UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Autonomicidade em uma rede definida por software utilizando teoria do perigo

Dissertação de Mestrado

Pablo Marques Menezes

São Cristóvão – Sergipe

2018

UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Pablo Marques Menezes

Autonomicidade em uma rede definida por software utilizando teoria do perigo

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de mestre em Ciência da Computação.

Orientador(a): Prof. Dr. Ricardo José Paiva de Britto Salgueiro

São Cristóvão – Sergipe

2018

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL
UNIVERSIDADE FEDERAL DE SERGIPE

Menezes, Pablo Marques
M543a Autonomicidade em uma rede definida por sftware utilizando teoria do perigo / Pablo Marques Menezes ; orientador Ricardo José Paiva de Britto Salgueiro. - São Cristóvão, 2018.
66 f., il.

Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Sergipe, 2018.

1. Ciência da computação. 2. Programas de computador - Testes. 3. Redes de computadores. 4. Engenharia de software. 5. Software - Testes. 6. . I. Salgueiro, Ricardo José Paiva de Britto, orient. II. Título.

CDU 004.4

Pablo Marques Menezes

Autonomicidade em uma rede definida por software utilizando teoria do perigo

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de mestre em Ciência da Computação.

Trabalho aprovado. São Cristóvão – Sergipe, 30 de Agosto de 2017:

Prof. Dr. Ricardo José Paiva de Britto Salgueiro
Orientador

Prof. Dra. Edilayne Meneses Salgueiro

Prof. Dra. Fernanda Maria Ribeiro de Alencar

São Cristóvão – Sergipe
2018

Eu dedico esse trabalho a toda a minha família, amigos e professores que me deram o apoio necessário para chegar aqui.

Agradecimentos

À minha Família, especialmente minha esposa, Adelbla, e meus queridos filhos, Letícia e Miguel, pelo apoio e compreensão pela minha ausência durante esta jornada; à minha mãe Maria Luiza e meus bisavôs pela criação, carinho e amor, além de terem moldado meu caráter e de possibilitar uma educação de qualidade; aos meus amigos pela torcida e, em especial, a Fabio Rocha pelo incentivo ao mundo da pesquisa; aos colegas de trabalho do SENAI, Banese e UNIT; a todos os meus professores e colegas do Programa de Pós-Graduação em Ciência da Computação (PROCC); ao meu orientador Prof. Ricardo Salgueiro que, pelo voto de confiança e toda paciência dedicada, tornou este trabalho possível; e a Deus, por ter colocado em meu caminho todas as pessoas que fazem parte da minha vida.

Só sei que nada sei por completo
Só sei que nada sei que só eu saiba
Só sei que nada sei que eu não possa vir a saber
Só sei que nada sei que outra pessoa não saiba
Só sei que nada sei que eu e outra pessoa não saibamos juntos
(Mário Sérgio Cortella)

Resumo

Os data centers evoluíram em cenários cada vez mais complexos, tornando a gerência da rede uma tarefa difícil para os administradores, sobretudo no aspecto da segurança da informação. A necessidade de tornar os ambientes computacionais autonômicos é evidente devido à complexidade e a onipresença da tecnologia em quase todos os aspectos da vida humana. Além de trazer dinamismo aos negócios e novos serviços para os usuários, também traz riscos e complexidade na gestão. Neste cenário, várias pesquisas têm sido realizadas em busca de métodos a tornar estas complexas redes autogerenciáveis. Inspirado nos conceitos de redes autonômicas e no sistema imunológico humano, este trabalho utiliza algoritmo das células dendríticas no modelo de gerenciamento MAdPE-K e as características de programabilidade, gestão centralizada e descentralização dos planos de dados e controle das redes definidas por software para prover autonomicidade. Considerando que a maioria dos ataques a uma rede de computadores inicia-se com o reconhecimento dos ativos, nos experimentos utilizou-se ataques de varredura de portas Port Scan como evento anômalo. Este tipo de ataque foi utilizado para comprovar a eficácia da detecção do processo anômalo com abordagem das células dendríticas em um host. Nos experimentos é seguido todo o ciclo do modelo MAdPE-K e os resultados de reação foram considerados satisfatório, com tempo médio de 1,2 segundos entre a detecção do evento anômalo e reação com isolamento da origem do ataque.

Palavras-chave: SDN, MAdPE-K, Células dendríticas.

Abstract

Data centers have evolved in increasingly complex scenarios, making network management a difficult task for administrators, particularly in the area of information security. The need to make autonomous computing environments evident is due to the complexity and ubiquity of technology in almost every aspect of human life. In addition to bringing business dynamism and new services to users, it also brings risks and complexity in management. In this scenario, several types of research have been carried out in search of methods to make these complex networks self-manageable. Inspired by the concepts of autonomic networks and the human immune system, this work uses dendritic cell algorithm in the MAdPE-K management model and the characteristics of programmability, centralized management, and decentralization of data planes and control of software-defined networks to provide autonomy. Considering that most attacks to a computer network start with the recognition of the assets, in the experiments Port Scan port scanning attacks were used as an anomalous event. This type of attack was used to prove the efficacy of detecting the anomalous process with the approach of the dendritic cells in a host. In the experiments, the whole cycle of the MAdPE-K model was followed and the reaction results were considered satisfactory, with an average time of 1.2 seconds between the detection of the anomalous event and the reaction with the isolation of the origin of the attack.

Keywords: SDN, MAdPE-K, Dendritic cells.

Lista de ilustrações

Lista de tabelas

Lista de códigos

Lista de abreviaturas e siglas

AIS	Artificial immune system
CERT-BR	Centro de estudos e respostas e tratamento de incidentes de segurança do Brasil
CSM	Costimulatory Molecules
DCA	Dendritic cells algorithm
dDCA	Deterministic dendritic cells algorithm
DDoS	Distributed Denial of Service
DoS	Denial of Service
DT	Danger Theory
HIS	Human Immune System
IS	Inflammation Signal
JSON	JavaScript Object Notation
MCAV	Mature Context Antigen Value
MAdPE-K	Monitor-Analyze danger-Plan-Execute plus Knowledge
MAdPE-K/SDN	MAdPE-K with Software-Defined Network
MAPE-K	Monitor-Analyze-Plan-Execute plus Knowledge
PAMP	Pathogen-associated molecular pattern
REST	Representational State Transfer
SDN	Software Defined Network
SIEM	Security Information and Event Management
SS	Sinal Seguro
SP	Sinal de Perigo
TICs	Tecnologia da Informação e Comunicação

Sumário

1 Introdução

A crescente quantidade de ameaças aos sistemas computacionais tornam a administração de uma infraestrutura de rede difícil e suscetível a falhas por erro humano. Segundo dados do CERT-BR (Centro de estudos e respostas e tratamento de incidentes de segurança do Brasil) foram relatados 647.112 incidentes reportados no ano de 2016. Tomar ações contra todos os ataques em tempo hábil é uma tarefa extremamente difícil para um administrador e mais perigosa se fizer necessário realizar mudanças nas configurações dos equipamentos eletrônicos, nas tecnologias e/ou na topologia envolvida na transmissão de dados. A computação autonômica surge para dotar os sistemas computacionais de características do sistema imunológico humano tornando-os autogerenciáveis. Kephart e Chess (2003) apresentaram uma arquitetura de um elemento autonômico com o modelo de gerência MAPE-K (*Monitor-Analyze-Plan-Execute plus Knowledge*) (IBM, 2005). O gerenciador autonômico tem a função de gerenciar quaisquer elementos, que podem ser qualquer ativo da rede.

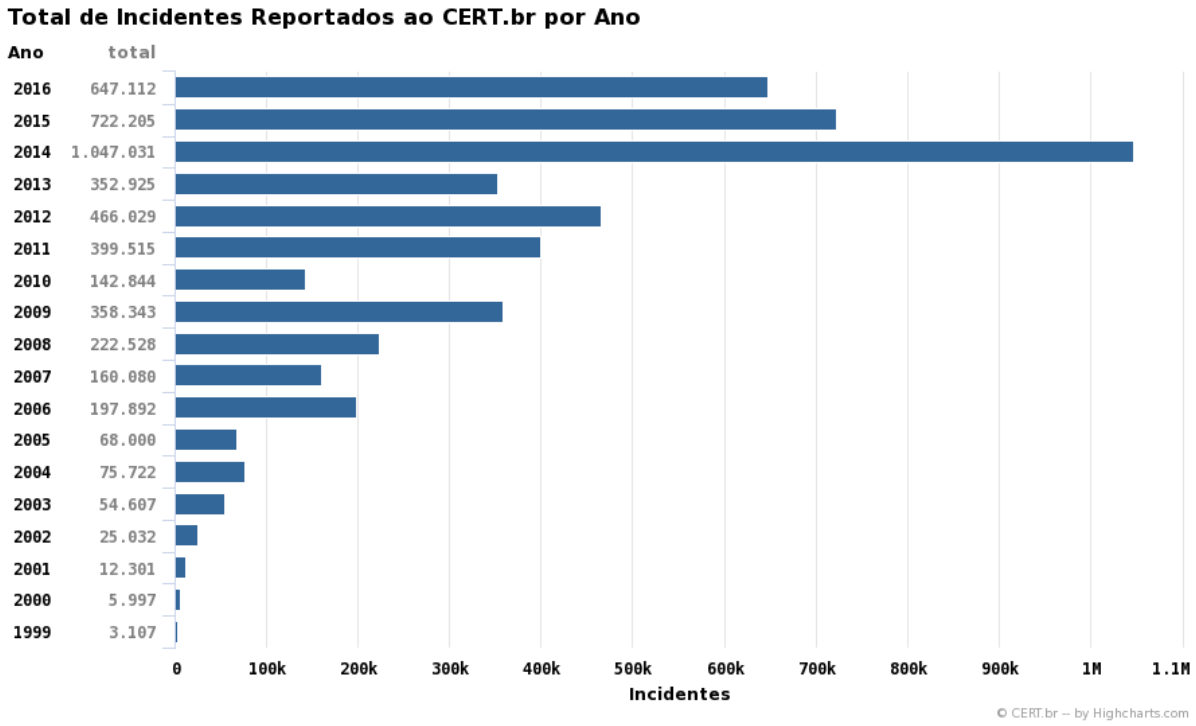
A Figura 1.1 ilustra a quantidade de incidentes reportados entre os anos de 1999 e 2016 ao CERT-BR, no ano de 2016. Desdes incidentes, 59,33% foram de varreduras em redes de computadores (*scan*), ataque que tem como objetivo coletar informações com intuito de identificar quais dispositivos estão ativos e quais serviços estão sendo disponibilizados. Este tipo de ataque é amplamente utilizado na etapa de reconhecimento, primeira etapa de um ataque, pois a identificação de potenciais alvos permite associar possíveis serviços habilitados às vulnerabilidades existentes. Os tipos de ataques relatados ao CERT-BR em 2016 estão apresentados na Figura 1.2.

A coexistência de redes heterogêneas em uma infraestrutura necessita de um novo paradigma para administrar e monitorar comportamentos maliciosos visto que as ameaças não estão mais confinadas em seus domínios de rede de origem (HASHIM; MUNASINGHE; JAMALIPOUR, 2010). Para este cenário, pesquisas feitas para dotar as redes de características

do sistema imunológico humano (*Human Immune System - HIS*) possibilitam o desenvolvimento de mecanismos de proteção dando origem aos sistemas imunes artificiais (*Artificial immune system - AIS*) (CASTRO; ZUBEN; KNIDEL, 2007).

Os estudos que propuseram o uso de características do sistema imunológico aplicado à computação datam de 1986 e várias teorias foram trabalhadas ao longo de tempo como seleção negativa, seleção clonal e teoria do perigo (YANG et al., 2014).

Figura 1.1 – Histórico de incidentes reportados ao CERT-BR.



Fonte: CERT-BR

A utilização de sistemas imune artificiais com o intuito de prover segurança e detecção de ameaças am ambientes computacionais vem sendo amplamente estudados pela academia conforme tratado por Yang et al. (2014), Shamshirband et al. (2014) e Fernandes et al. (2017).

As abordagens com utilização da teoria do perigo para proteção de redes contra ataques de negação de serviço (*Denial of Service DoS*), como tratado por Rawat e Saxena (2009) e por Ahmad, Idris e Kama (2017), ou detecção escaneamento, conforme apresentado por Greensmith e Aickelin (2007) e refeito por Anandita et al. (2015) e Silva, Caminhas e Errico (2017), trazem resultados aplicáveis.

O sistema imunológico humano inspira outras áreas de estudo objetivando a segurança dos ativos dos sistemas computacionais, os sistemas artificiais imunes. Este trabalho baseia-se na teoria do perigo como estratégia para a detecção de ataques. A teoria do perigo é um modelo que utiliza abordagem com base em sinais que alertam níveis de perigo. Dentre as técnicas apresentadas na literatura destaca-se o algoritmo de células dendríticas, já utilizado

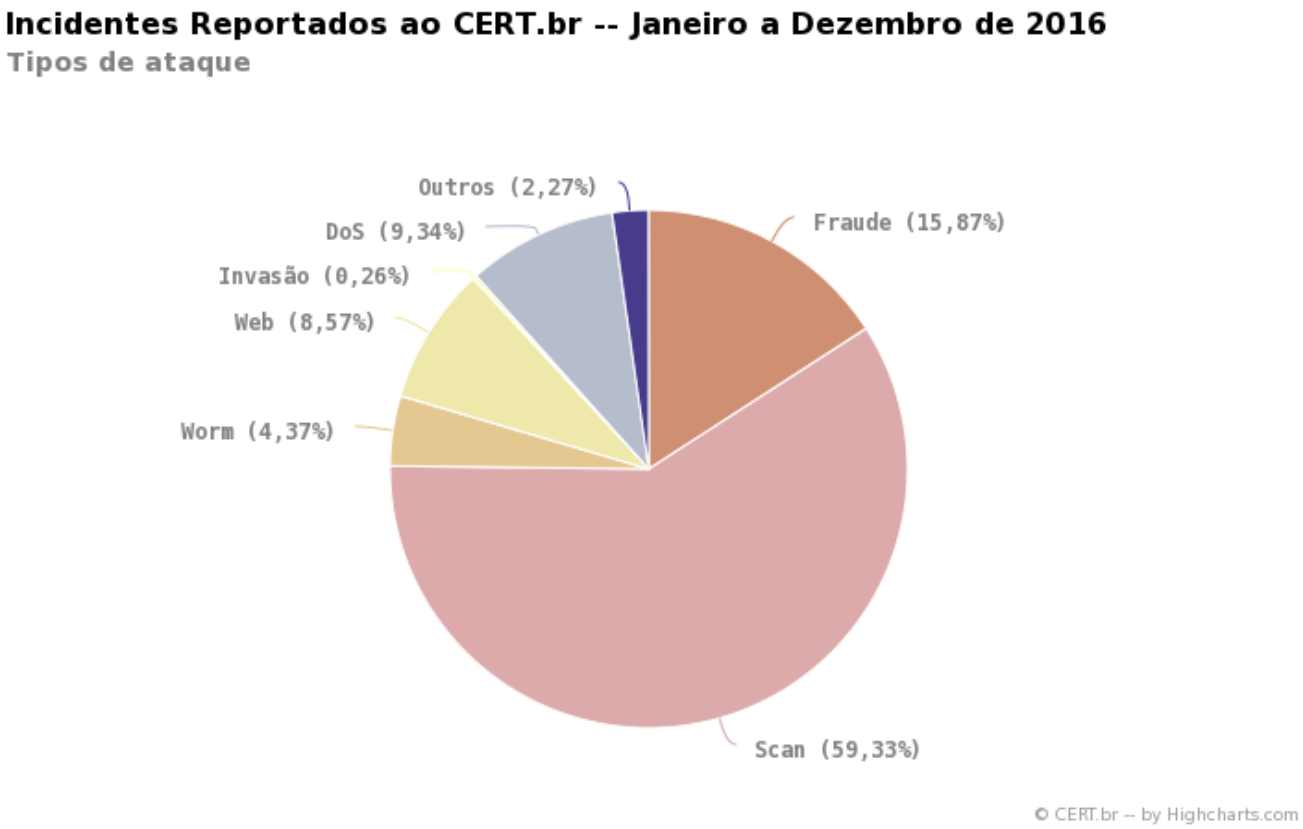
com resultados promissores (GREENSMITH; AICKELIN, 2007), (ANANDITA et al., 2015) e (OLIVEIRA; SALGUEIRO; MORENO, 2013).

As redes definidas por *software (Software-Defined Network -SDN)* mudam a maneira de gerenciar e administrar uma rede de computadores. Este novo paradigma possibilita, devido à sua programabilidade, melhoria nos tratamento de ameaças existentes como também a possibilidade de novas ameaças referentes aos elementos de uma rede SDN (YAN et al., 2015).

As redes definidas por software podem ser descritas como autogerenciáveis, desde que os elementos de autoconfiguração, autocura, auto-otimização e autoproteção possam ser fornecidos pelo controlador ao administrador através de uma interface (BEHRINGER et al., 2015).

Tendo em vista que há possibilidade dotar de autoproteção uma rede definida por software utilizando-se da abordagem teoria do perigo, este trabalho verifica a aplicabilidade destas abordagens fazendo uso da programabilidade das redes definidas por software, com base no modelo de alerta antecipado apresentado por Oliveira, Salgueiro e Moreno (2013) e tomando ação em tempo hábil para garantir a integridade da rede.

Figura 1.2 – Tipos de ataques reportados ao CERT-BR em 2016.



Fonte: CERT-BR

1.1 Problemática e Hipótese

Gerenciar a segurança de redes em uma infraestrutura cada vez mais complexa e heterogênea é uma tarefa árdua, pois é necessário uma monitoração contínua dos diversos ativos, analisar as mudanças no comportamento de cada um destes elementos e administrá-los adequadamente em caso de mudança de contexto. Decisões de gerenciamento devem muitas vezes promover mudanças devido a uma ameaça ou alteração na topologia, sem que comprometa o funcionamento da rede e dos serviços oferecidos.

Ameaça é a probabilidade de que haja uma exploração proposital ou acidental de alguma vulnerabilidade, há uma enorme variedade de ameaças que atingem hardware, software e rede objetivando comprometer a confidencialidade, integridade e disponibilidade das TICs (Tecnologia da Informação e Comunicação) (JANG-JACCARD; NEPAL, 2014).

Os ataques, técnicas ou ações que exploram uma vulnerabilidade, estão cada vez mais sofisticados. Jang-Jaccard e Nepal (2014) apresentam novos padrões de ataques em tecnologias emergentes, como as redes móveis, computação em nuvem, mídias sociais entre outras. Neste trabalho, os autores buscam explorar vulnerabilidades específicas destas tecnologias, colaborando para que a implantação e manipulação dos controles empregados para proteger a infraestrutura não sejam aplicados de maneira satisfatórias em todos os casos, gerando falsos positivos e falsos negativos, impedindo a detecção do ataque e por consequência dificultando a tomada de decisão para proteção.

Neste contexto, a busca por soluções de gerenciamento que possibilite a detecção prematura de ataques e reação em tempo hábil garantindo proteção efetiva da infraestrutura contra ataques é um problema aberto na literatura e explorado nesta dissertação.

Como hipótese para problemática aqui relatada tem-se, então, neste trabalho que é possível estabelecer um serviço autonômico para rede definida por software que proveja autogerenciamento e autoproteção com base na Teoria do Perigo (*Danger Theory - DT*), possibilitando a detecção, prevenção e atuação na mitigação dos ataques em redes e sistemas com eficácia, contribuindo de forma efetiva na gerência de segurança.

1.2 Objetivos

Este trabalho tem como objetivo prover um serviço de autogerenciamento e autoproteção, para detecção, prevenção e inibir a propagação de um ataque em rede definida por software.

Assim, como objetivo específico tem-se:

- Definição de um serviço de arquitetura que proveja o autogerenciamento às redes definidas por software para autoproteção;

- Implementar o algoritmo de células dendríticas para o estabelecimento de autonomicidade na rede;
- Adaptar soluções para as etapas de monitoramento, análise do perigo e execução, referentes ao modelo MAdPE-K.

1.3 Organização

Para melhor entendimento, este documento está estruturado em capítulos e seções, que são:

- Capítulo 2 - Fundamentação teórica: neste capítulo é apresentado os conceitos básicos das tecnologias e técnicas utilizadas nesta dissertação, divido em seções na seguinte ordem: na seção 2.2 são abordados os conceitos básicos, características e arquitetura das redes definidas por software ; na seção 2.3 é apresentado as definições e características de uma rede autonômica; na seção 2.4 esta seção aborda os conceitos da abordagem dos sistema autoimune baseado na teria do perigo e suas principais técnicas conforme literatura com foco no algoritmo de células dendrítica; na seção 2.5 é apresentado o estado da arte relacionados ao tema desta dissertação.
- Capítulo 3 - Arquitetura MAdPE-K/SDN: neste capítulo é feita a descrição da arquitetura proposta. As fases de monitoramento, análise do perigo, planejamento e execução da arquitetura MAdPE-K/SDN estão descritas nas seções 3.1, 3.2, 3.3 e 3.4 respectivamente.
- Capítulo 4 - Estudo de caso: é apresentado um estudo de caso com aplicação do sistema proposto, na seção 4.1 é apresentado ambiente utilizado para o experimento e na seção 4.2 é apresentada aplicação das fases da arquitetura proposta nesta dissertação e os resultados do experimento.
- Capítulo 5 - Considerações finais: neste capítulo trata das considerações finais, na 5.1 as contribuições desta dissertação e na seção 5.2 os trabalhos futuros.

2

Fundamentação teórica

Nas seções que seguem são apresentados os fundamentos e conceitos das redes definidas por software, rede autonômica e teoria do perigo. Tecnologias e técnicas base desta dissertação.

2.1 Segurança de redes

Manter ambientes computacionais seguros é uma tarefa árdua para os administradores e tendo seus serviços cada vez mais expostos na Internet, maiores são os riscos. Desde modo, há uma busca no estado da arte para garantir a disponibilidade, integridade e confidencialidade.

Conforme descrito em International Organization for Standardization (2014), ataque é uma tentativa de destruir, expor ou obter acesso não autorizado ou fazer uso deste. Os ataques para alcançar sucesso na investida são, de modo geral, executados em 4 etapas básicas (BHUYAN; BHATTACHARYYA; KALITA, 2011):

1. **Coleta de informação**: nesta etapa, o foco do atacante é descobrir o máximo de informações de vulnerabilidade da rede e assim utilizar algumas informações que possam ser usadas no ataque.

2. **Avaliação de vulnerabilidades**: após a coleta de informações, o invasor avalia as informações encontradas para planejar o ataque e tenta comprometer alguns dos nós na rede, utilizando códigos maliciosos, como um precursor do lançamento de ataques.

3. **Exploração**: nesta etapa o ataque é iniciado para atingir o alvo a partir dos nós já comprometidos.

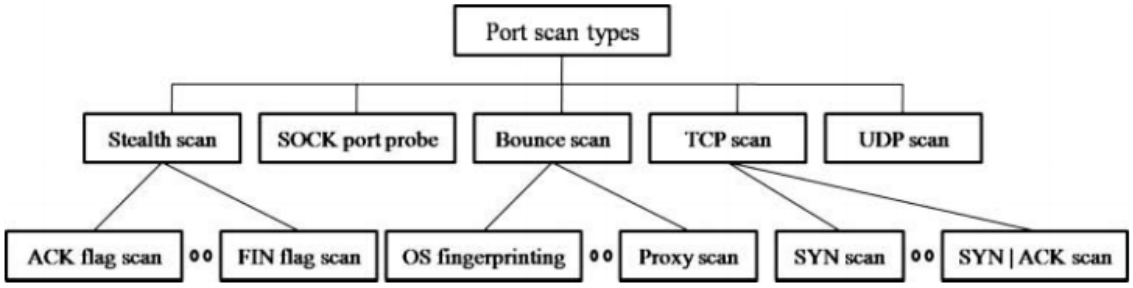
4. **Limpeza**: Por fim, o atacante remove todos os arquivos de registro (logs) na tentativa de eliminar o histórico de ataques.

Na etapa de coleta de informações, uma das técnicas utilizadas para o reconhecimento é o de varredura de portas *(port scans)*. Esta técnica é utilizada pelos invasores na detecção de nós ativos e os seus serviços em atividade. A identificação dos serviços é feita a partir dos dados presentes nos campos de cabeçalho dos protocolos da camada de transporte *TCP e UDP*.

Há dois tipos básicos de varredura de portas, a discreta que não estabelece uma comunicação completa com o alvo, diferentemente da varredura de força bruta que estabelece uma comunicação completa e tem como característica uma quantidade de pacotes do tipo *SYN* para todas as portas de um mesmo nó. Alguns dos padrões de varredura podem ser identificados como *SYN SCAN, ICMP SCAN, ACK SCAN, UDP SCAN*, não se limitando só a esses (GADGE; PATIL, 2008).

Bhuyan, Bhattacharyya e Kalita (2011) apresentam os tipos de técnicas de *port scan*, representado na Figura 2.1. Essa técnica consiste em enviar mensagens para cada uma das 65536 portas disponíveis. Quando há retorno de alguma tipo dessas mensagens significa que há indicação de algum serviço ativo.

Figura 2.1 – Tipos de *port scan*.



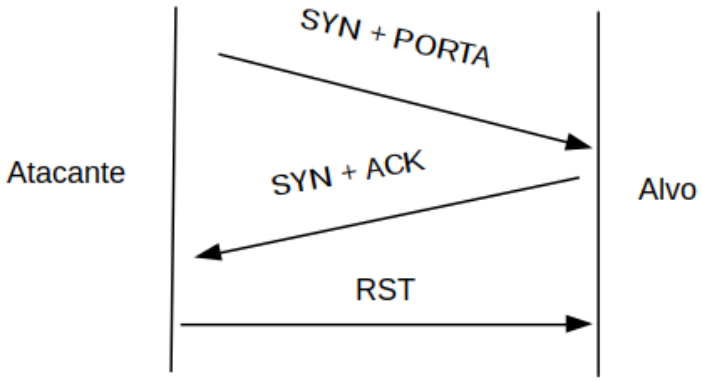
Fonte: (BHUYAN; BHATTACHARYYA; KALITA, 2011)

O padrão de varredura *SYN Scans* é muito utilizado na identificação de alvos e tem como base o protocolo TCP. Este padrão funciona enviando um grande número de pacotes com apenas o pacote do tipo *SYN* para o alvo. Esta verificação não completa o *handshake* que estabelece conexão TCP em 3 vias e encerra a conexão, enviando um pacote com mensagem *RST*, depois que a vítima responde com uma pacote com mensagem *SYN/ACK* indicando uma porta aberta. Esta varredura pode ser facilmente identificada se houver um grande número de pacotes com o sinalizador *SYN* definido neles provenientes de um único *host*. Esta técnica está representada pela Figura 2.2.

Uma ferramenta de código aberto para exploração de redes e de maneira eficaz é o *NMAP Security Scanner*, usada por uma grande parcela dos interessados em realizar reconhecimento de nós e serviços em uma rede (LYON, 2009).

Inibir ataques do tipo *SCAN* é altamente relevante pois, sendo evitados, elimina ou dificulta a continuação das etapas posteriores. Assim, há uma gama de estudos que tratam da

Figura 2.2 – *Port scan* em ação.



Fonte: Autor

detecção deste ataque, conforme apresentado por Bhuyan, Bhattacharyya e Kalita (2011).

2.2 Redes definidas por software

O esforço de tornar as redes de computadores programáveis foram iniciados desde a década de 90, nesse período destaca-se o primeiro projeto para separação dos planos de dados e controle o Tempest. Posteriormente, entre 2000 e 2008, outros projetos foram desenvolvidos, o ForCes e Ethane, respectivamente. Até que, em 2008, OpenFlow surgiu e, juntamente com os projetos de virtualização de rede como *Mininet* e *Open vSwitch*, tornaram SDN uma realidade (FEAMSTER; REXFORD; ZEGURA, 2014) e (SEZER, 2013).

SDN é um novo paradigma de rede que segundo, a *Open Networking Foundation (ONF)* (FOUNDATION, 2012), surgiu para resolver os principais problemas existentes no modelo de redes tradicionais. No modelo atual, o dispositivo de rede recebe configurações para uma determinada função específica tornando dispendioso e complexo alterações de política da rede ou localidade dos dispositivos e impossibilitando alterações dinâmicas na topologia da rede para atender alguma necessidade.

Outro problema apontado pela ONF (FOUNDATION, 2012) é a incapacidade das redes atuais referentes à escalabilidade em acompanhar as demandas de processamento paralelo, *Big Data* e armazenamento em rede. Apesar destes problemas a ONF aponta que há uma coordenação e cooperação entre os dispositivos, estes devem ser primordialmente dos mesmos fornecedores, gerando dependência. O novo paradigma centraliza o gerenciamento e controle de dispositivos de vários fornecedores, com uso de APIs (*Applications Programming Interface*) para gerenciamento e automação. Isso representa uma rápida inovação devido a capacidade de fornecer novos recursos sem dependências dos fornecedores ou configuração individuais, gestão centralizada e automatizada dos dispositivos e aplicação de políticas de rede de forma uniforme, assim

reduzindo as falhas de configuração.

SDN é apresentado como um paradigma de rede que poderá romper com as limitações das redes atuais, pois quebra a integração vertical do modelo de rede atual, separando o plano de controle do plano de dados. No modelo atual estes planos estão embutidos nos dispositivos de rede, ou seja, as regras de como os pacotes serão encaminhados e criação de tabelas de roteamento e sua execução estão em todos os dispositivos. Assim no novo paradigma há um software central, contido no plano de controle, que controla o comportamento da rede e tem responsabilidade de administrar e gerenciar as regras de encaminhamento, fazendo com que os dispositivos, contidos no plano de dados que é responsável pelo encaminhamento dos pacotes, atuarem de maneira mais simples (KIM; FEAMSTER, 2013). Este *software* central, neste novo paradigma é localizado em um novo elemento, denominado controlador.

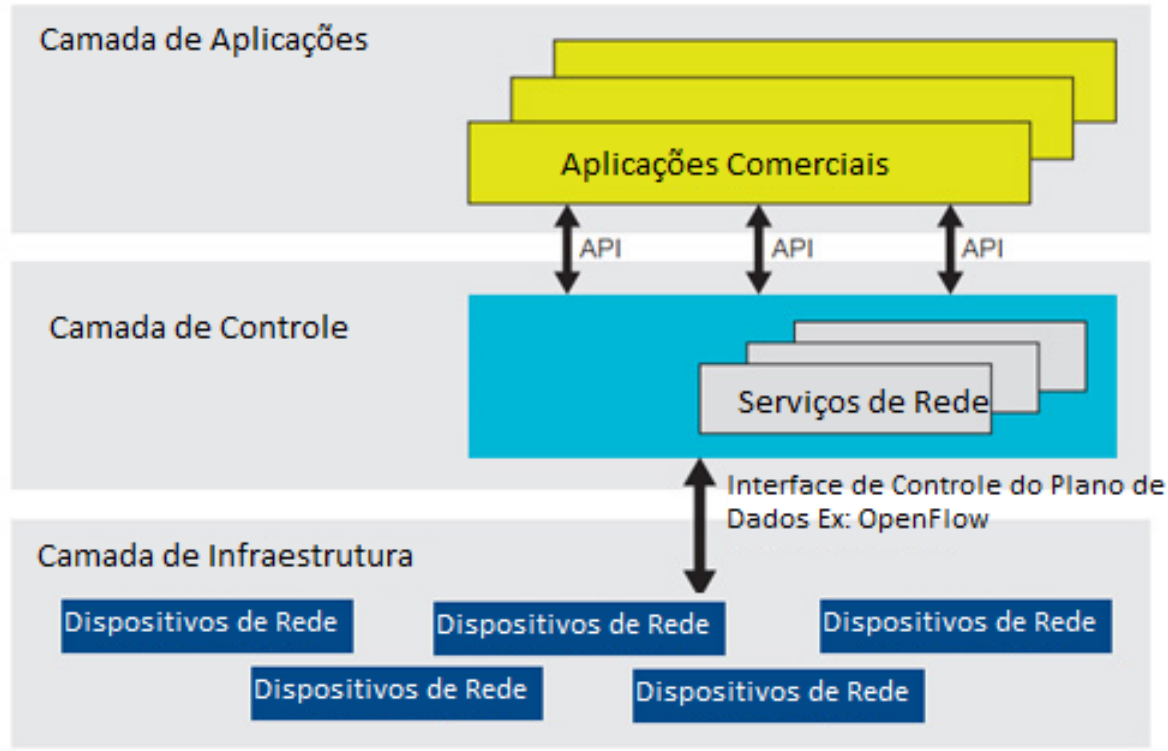
A Figura 2.3 apresenta a arquitetura básica da SDN, dividida de forma vertical e horizontal, em 3 camadas (KREUTZ D.; RAMOS; VERISSIMO; ROTHENBERG C.AND AZODOLMOLKY, 2015):

- **Camada de aplicação (*Application Layer*)**: esta camada é a mais alta e trata com aplicativos finais que utilizam os serviços SDN.
- **Camada de Controle (*Control Layer*)**: nesta camada estão alocados o conjunto de controladores que permitem gerenciar as políticas dos dispositivos da camada física.
- **Camada de Infraestrutura (*Infrastructure Layer*)**: esta camada compreende os dispositivos do plano de dados como os *switches*.

O controlador é a parte principal do SDN que funciona entre dispositivos de rede e várias aplicações que fornecem uma interface programática para a rede. Um controlador SDN encontra-se na camada de controle e tem a responsabilidade exclusiva de gerenciar os protocolos de rede, as políticas e estabelecer o caminho na rede. Há várias implementações de controladores SDN atualmente, Khondoker et al. (2014) fizeram uma análise dos 5 controladores mais relevantes na literatura: POX (The POX Controller Team, 2016), Ryu (RYU SDN Project Team, 2016), Trema (The Trema Controller Team, 2016), FloodLight (FLOODLIGHT, 2014) e OpenDaylight (OPENDAYLIGHT, 2015). Nesta dissertação, as soluções desenvolvidas foram implementadas utilizando o controlador RYU. Esta escolha baseou-se nos resultados obtidos em Khondoker et al. (2014), que teve o controlador RYU como destaque.

Na Figura 2.3 está apresentada as camadas da SDN, e quais os elementos que compõem cada camada. A comunicação entre as camadas é realizada pelas interfaces de comunicação com as camadas superiores (*Northbound*) e inferiores (*Southbound*), a localização destas camadas de abstração são mostradas na Figura 2.4 de acordo com a RFC 7426 apresentada por E. Haleplidis et al. (2015).

Figura 2.3 – Camadas SDN.



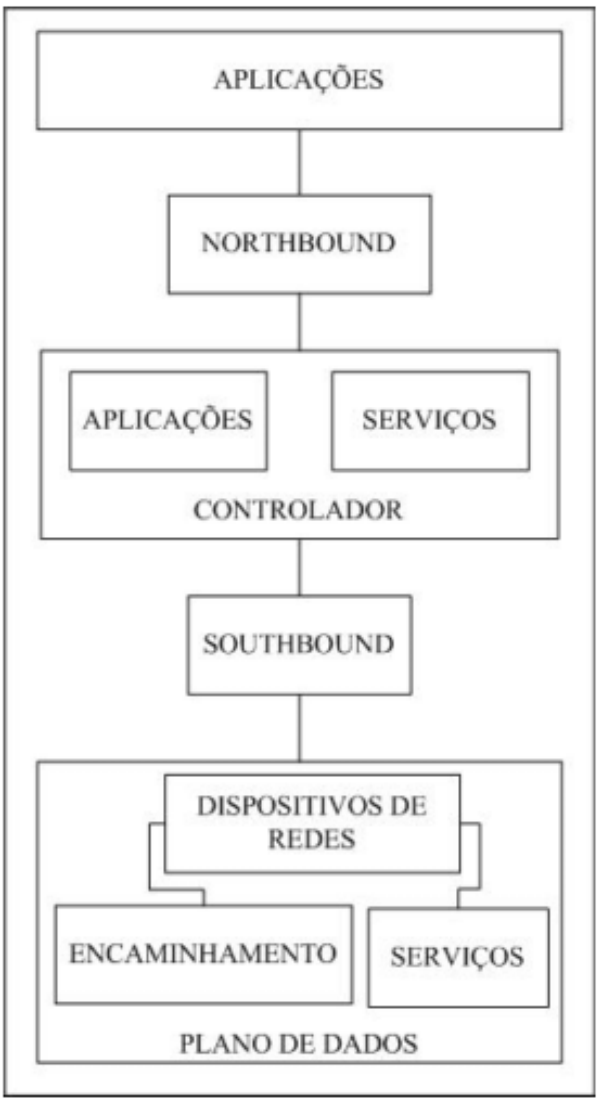
Fonte: (FOUNDATION, 2012)

Este novo paradigma possui benefícios conforme apresentado por Sahoo et al. (2016):

- Uso eficiente dos recursos;
- Eficiência na administração da rede;
- Otimização na comunicação cruzadas dos inquilinos em um *data center*;
- Programabilidade da rede;
- Gerenciamento de rede tanto física quanto virtual;
- Redução de custos;
- Gerenciamento centralizado de rede;
- Segurança aprimorada.

A Figura 2.5 ilustra as funções das camadas, evidenciando os benefícios citados e as interações entre as camadas. Além de apresentar a localização de algumas soluções e tecnologias utilizadas, na arquitetura SDN.

Figura 2.4 – Arquitetura da SDN

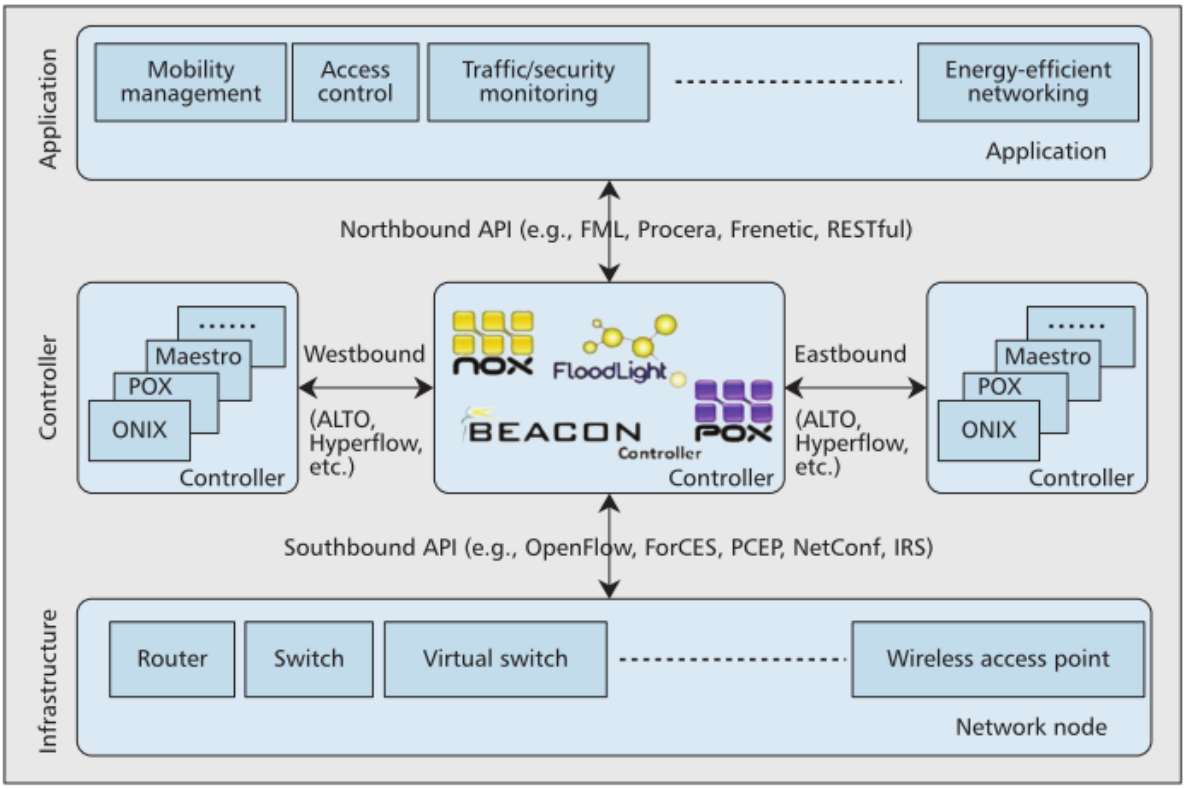


Fonte: (BOMFIM, 2017)

O novo paradigma ao promover flexibilidade e programabilidade, possibilita mudanças necessárias e eficientes na rede no auxílio da mitigação de ameaças, entretanto a separação dos planos de controle do plano de dados pode fragilizar a segurança da rede. Atacantes podem tirar vantagem da característica da centralização da SDN e lançar ataques contra a camada de controle, camada de infraestrutura e a camada de aplicação da SDN.

Vários estudos vem sendo feitos para levantar o estado da arte no que diz respeito a segurança em redes SDN, utilizando-as para mitigação de ameaças em rede tradicionais e/ou das novas ameaças surgidas nesse novo paradigma (NUNES et al., 2014), (SCOTT-HAYWARD; O'CALLAGHAN; SEZER, 2013), (SCOTT-HAYWARD; NATARAJAN; SEZER, 2016) e (BOMFIM et al., 2017).

Figura 2.5 – Tecnologias e Ferramentas da SDN



Fonte:(SEZER et al., 2013)

2.3 Redes Autonômicas

Em 2001, Horn (2001 apud KEPHART; CHESS, 2003) apresentou a ideia de comutação autonômica como sendo a opção para suprir as necessidades de alto nível dos administradores o termo relacionado ao sistema nervoso autônomo que gerencia todas as funções de baixo nível.

A definição de uma rede autonômica está descrito por Behringer et al. (2015) na RFC 7575 como sinônimo de autogestão, ou seja adapta-se às mudanças no ambiente sem intervenção de entidades externas, mas que permite orientação de alto nível para aplicar as políticas definidas pelos administradores, usadas para operar a rede.

A RFC 7575 ainda trás outras definições de uma rede autônoma apresentadas a seguir (BEHRINGER et al., 2015):

- **Intenção**: fornecida por uma unidade central e tem como escopo um domínio autônomo pois não há uma configuração ou informação para um nó específico e pode ser definido como uma política de operação da rede;
- **Domínio autonômico**: uma coleção de nós autônomos que instanciam uma mesma intenção;

- **Função autonômica**: um recurso ou função que não exige configuração e pode derivar todas as informações necessárias através de auto-conhecimento, descoberta ou intenção.
- **Agente de serviço autonômico**: implementado em um nó autônomo que executa uma função autonômica.
- **Nó autonômico**: emprega funções exclusivamente autônomas.
- **Rede autonômica**: uma rede que contém nós exclusivamente autônomos podendo conter um ou mais domínios autônomos.

O objetivo fundamental dos sistemas autonômicos é a autogestão, ou seja, ter o mínimo de intervenção humana na administração do sistema. As característica de um sistema autonômico é a autoconfiguração, auto-otimização, autocura e autoproteção (BEHRINGER et al., 2015).

A autoproteção em sistemas autonômicos possui duas frentes: defender o sistema como um todo e antecipar problemas (KEPHART; CHESS, 2003). Mais propriamente, a defesa contra problemas relacionados a ataques maliciosos, ou falhas em cascata não corrigidas por medidas de autocura, bem como a antecipação dos problemas com base nos dados dos sensores e assim tomar medidas para evitá-los ou mitigá-los.

A Tabela 2.1 apresenta um comparativo dos aspectos de autogestão em um ambiente computacional tradicional e um autônomo.

Tabela 2.1 – Aspectos de auto gestão: Computação Atual *versus* Computação Autonômica

Conceito	Computação Tradicional	Computação Autonômica
Auto-configuração	Os centros de dados corporativos possuem vários fornecedores e plataformas. Instalar, configurar e integrar sistemas é demorado e propenso a erros.	A configuração automatizada de componentes e sistemas segue políticas de alto nível. O resto do sistema se ajusta de forma automática e perfeita.
Auto-organização	Os sistemas têm centenas de parâmetros de ajuste não lineares configurados manualmente e seu número aumenta com cada versão.	Componentes e sistemas continuamente buscam oportunidades para melhorar seu próprio desempenho e eficiência.
Auto-cura	A determinação de problemas em sistemas grandes e complexos pode levar uma equipe de semanas de programadores.	O sistema detecta, diagnostica e repara automaticamente problemas localizados de software e hardware.
Auto-proteção	A detecção e recuperação de ataques e falhas em cascata é manual.	O sistema defende automaticamente ataques mal-intencionados ou falhas em cascata. Ele usa aviso prévio para antecipar e prevenir falhas no sistema.

Fonte: (KEPHART; CHESS, 2003)

Tendo como base as definições apresentadas na RFC 7575 por Behringer et al. (2015), uma rede definida por software possui elementos que podem, caso seus elementos forneçam alguma das propriedades que garantem uma ou mais funções autonômicas, ser descrita como autogerenciável.

O modelo de referência autonômica descrito em Behringer et al. (2015) apresenta os elementos contido em um nó autonômico que são:

- **Agentes de serviços autonômicos**: eles implementam o comportamento autonômico de um serviço ou função específica.
- **Autoconhecimento**: um nó autonômico conhece suas próprias propriedades e capacidades;
- **Conhecimento de rede**: um agente de serviço autônomo pode requerer várias funções de descoberta na rede;
- ***Loops* de *Feedback***: elementos de controle fora do nó podem interagir com os nós autônomos;
- **Um agente de usuário autônomo**: fornece um *front-end* para usuários externos, permitindo o monitoramento da rede autônoma;
- **Plano de controle autônomo**: permite que o nós comuniquem-se entre si.

A arquitetura apresentada na Figura 2.6 mostra as fases do ciclo de gerenciamento: Monitoramento, Análise, Planejamento e Execução. O ciclo estão contido no gerenciador e os elementos gerenciados estão representados, podendo a partir de um conhecimento, segundo o modelo MAPE-K (KEPHART; CHESS, 2003) ser qualquer recurso da rede.

A Figura 2.7 ilustra os componentes essenciais para o elementos gerenciados. O sensor tem como finalidade a coleta de dados e os executores (*effector*) são os responsáveis pela ações necessárias no estado do elemento.

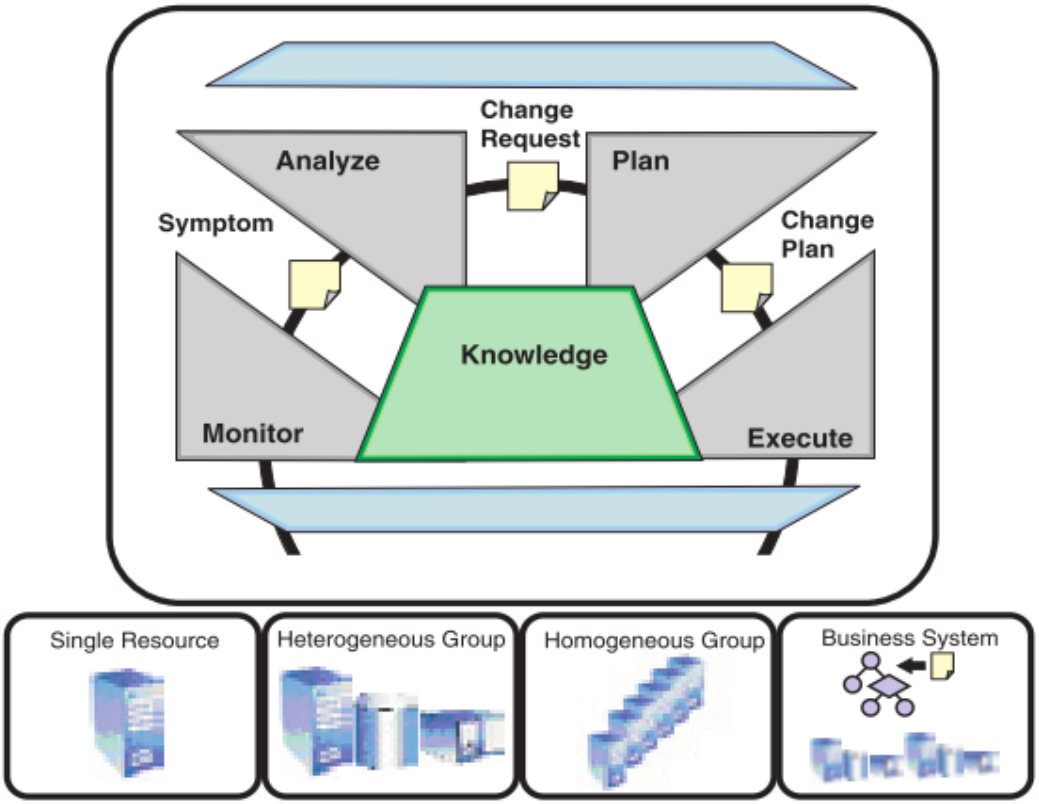
2.4 Teoria do Perigo

Os mecanismos de defesa contra agentes externos são inerentes ao seres vivos, protegendo o organismo, reagindo de forma rápida e eficaz. Devido a essa performance, o sistema imunológico humano vem sendo pesquisado para ser aplicado em sistemas computacionais com o mesmo propósito de reagir de forma rápida e eficaz a eventos anômalos. Para entender o funcionamento do sistema imunológico humano vários trabalhos foram desenvolvidos.

As primeiras abordagens estavam baseadas na grande capacidade de reconhecimento de padrões para distinguir entre o que era ou não próprio do organismo. Tal abordagem definida como seleção negativa, mantém circulando o organismo que não tem seus padrões identificados pelo linfócitos, já os organismos cujo padrões são identificado pelo linfócito, são eliminados (MATZINGER, 1994).

Uma segunda teoria denominada Teoria do Perigo (*Danger Theory - DT*) reduz as ocorrências de falsos positivos e negativo da abordagem de próprio e não próprio, pois aumenta

Figura 2.6 – Arquitetura de um nó autônomo com modelo Mape-K.



Fonte: (IBM, 2005)

sua capacidade de distinção com base em elementos e eventos que produzem perigos. A Teoria do perigo tem como base a ideia de que uma resposta imune é ativada na ocorrência de um dano à célula devido ao comportamento de células imunes chamadas de células dendríticas (*Dendritic Cells - DCs*).

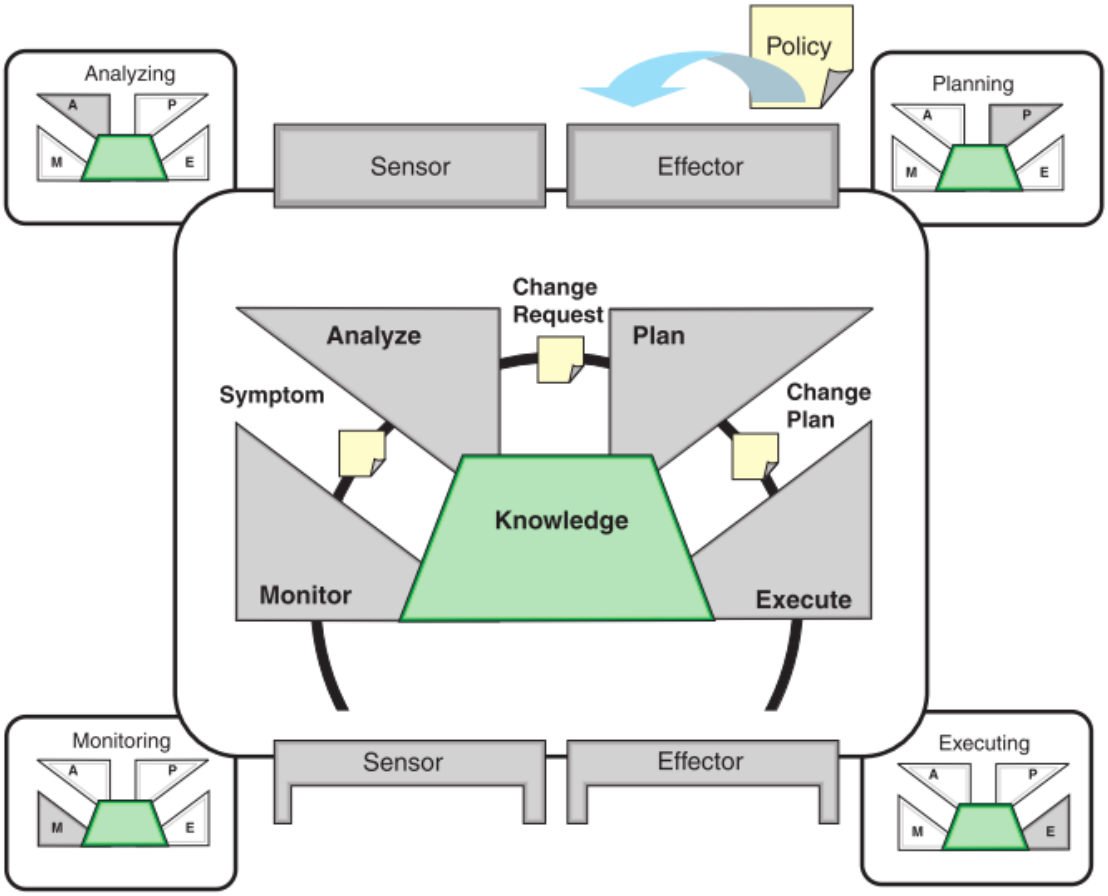
Assim, Matzinger (1994) propôs a Teoria do Perigo na tentativa explicar o funcionamento das respostas imunes do corpo humano, contrapondo a visão da seleção negativa, afirmando que a resposta imune tem origem nas reações aos estímulos que o ambiente reconhece como dano.

A teoria do perigo propõe que a morte de um tecido saudável dá-se de forma controlada, fenômeno conhecido como apoptose. Quando há apoptose as moléculas imunossupressoras são liberadas indicando que não há eventos de perigo, ou seja enviam sinais seguros. Por outro lado quando há morte celular de forma não natural, necrose, sinais de perigo são liberados no tecido, tornando o sistema imunológico sensíveis à mudanças na concentração de sinais de perigo.

A Figura 2.8 mostra o fenômeno de morte celular por apoptose e necrose e a ação da célula dendrítica.

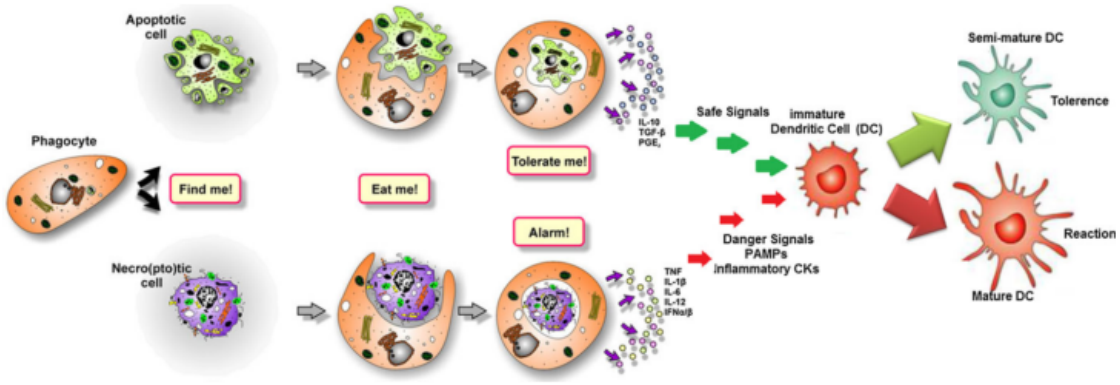
As DCs têm a responsabilidade de capturar, revelar e transformar antígenos nas células T, responsáveis por reconhecer corpos estranhos no organismo e de combatê-los (MATZINGER,

Figura 2.7 – Funcionamento em detalhes de gerente autonômico.



Fonte: (IBM, 2005)

Figura 2.8 – Ação de uma célula dendrítica e apoptose versus necrose



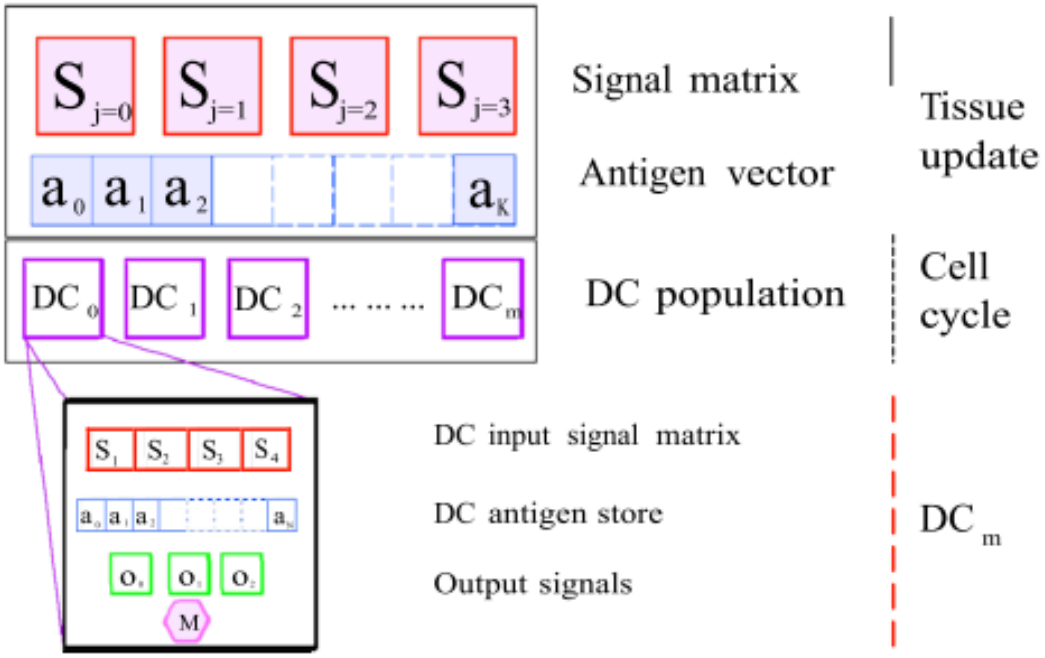
Fonte: (CHELLY; ELOUEDI, 2016)

1994), além de receber sinais vindos da vizinhança. Existem quatro tipos principais de sinais:

- **Sinais PAMP (PAMP - *Pathogen Associated Molecular Pattern*)**: produzidos por moléculas associadas a grupos de agentes patogênicos. Quando estes sinais são recebidos pelas DCs produzem moléculas coestimulatórias (CSM), que representa uma indicação que há um elemento estranho no organismo.
- **Sinais seguros (SS)**: são indicadores de uma situação normal, morte natural da célula (apoptose). Ou seja, o antígeno coletado pelas DCs não é prejudicial e não há necessidade de uma reação imune.
- **Sinais de perigo (SP)**: são indicadores de uma situação anormal, morte não natural da célula (necrose). Possuem menor valor de confiabilidade que os sinal PAMP.
- **Citocinas inflamatórias ou sinais de inflamação (*IS - Inflammation Signal*)**: são sinais que têm o efeito de amplificar os outros três sinais.

O grau de influência dos sinais dá-se através da distribuição de pesos, conforme ilustrado na Tabela 2.2. O peso de maturação do sinal seguro (SS) tem o valor negativo, pois é responsável para prevenir falsos positivos, já que tem efeito supressor nos sinais de PAMP e de perigo. O processamento dos sinais é realizado pela Equação 2.1, gerando três sinais de saída: CSM, células semimaduras e maduras.

Figura 2.9 – Componentes de uma aDC



Fonte: (GREENSMITH; AICKELIN; TWYCROSS, 2006)

Tabela 2.2 – Sinal de saída com base nas distribuição de pesos dos sinais.

W	CSM	semi	mat
PAMPs	2	0	2
Sinal de Perigo	1	0	1
Sinal Seguro	2	3	-3

Na figura 2.9 estão representadas as estrutura de dados do algoritmo de células dendríticas (aDC). A cada ciclo de vida é calculada utilizando a fórmula 2.1 com base nos valores da tabela 2.2.

$$O_j = \sum a_{i=0}^{n}(W_{ij} * S_i) * (1 - IC)_j \tag{2.1}$$

onde, com base na Tabela 2.2, O_j é o sinal de saída de índice j podendo ter os valores 0 (CSM), 1 (semimaduras) ou 2 (maduras); i é o índice da categoria do sinal de entrada podendo ter os valores 0 (sinal de pamp), 1 (sinal de perigo) ou 2 (sinal seguro); n é o número de categorias de sinais de saída menos um; W_{ij} é o peso da categoria de sinal de entrada i para o cálculo do sinal de saída O_j; S_i o número de sinais de entrada de categoria i; e IC o coeficiente de inflamação. Quando a soma dos valores de CSM calculados por cada aDC ultrapassa um limiar de migração, a aDC muda seu estado e se torna madura.

De acordo com a intensidade dos sinais é possível estabelecer o MCAV (*Mature Context Antigen Value*). O MCAV é um índice de anomalia de um antígeno calculado a partir da média ponderada das células maturadas com o total de células migradas. Seu valor é definido em um intervalo entre 0 e 1, sendo 0 o valor caso indique uma situação possivelmente normal e 1 a possibilidade de ocorrência de um evento anômalo (GREENSMITH; AICKELIN, 2007).

As DCs podem ter três comportamentos distintos, como pode ser visto na Figura 2.8, a depender da concentração dos sinais recebidos da seguinte forma (GREENSMITH; AICKELIN; CAYZER, 2005):

- Estado imaturo é o estado inicial da DC, ou seja, sem concentração de sinais recebidos;
- Estado semimaduro é o estado quando a DC recebe maior concentração de sinais seguros.
- Estado maduro quando expostas a um maior quantidade de sinais PAMP ou de perigo em relação aos sinais seguros;

O comportamento das DCs serviu de modelo para o desenvolvimento de um algoritmo de classificação imunológica chamado algoritmo de células dendríticas (DCA) por Greensmith, Aickelin e Cayzer (2005).

O pseudocódigo 2.1 é uma versão determinística do algoritmo de células dendríticas apresentado por Greensmith e Aickelin (2008).

Código 2.1 – Pseudocódigo do Algorítimo das Células Dendríticas Determinístico (dDCA) (GREENSMITH; AICKELIN; TWYCROSS, 2006)

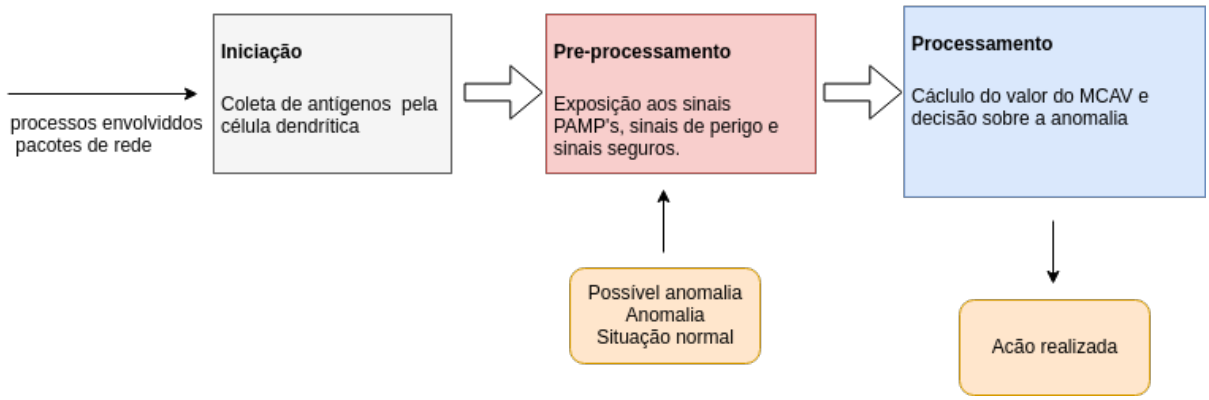
```
entrada : antigenos e sinais
saida : tipos de antigenos e Ka

define tamPopulacao de DCs;
inicializa DCs;
        enquanto houver dados de entrada
                escolha entrada
                        caso antigeno
                                antigenoContador++;
                                celulaIndice = agContador %= tamPopulacao;
                                DC->antigeno de indice cellIndex++;
                                atualiza o perfil do antigeno da DC;
                        fim
                        caso sinal
                        calcula csm e k;
                        para cada DC
                                DC.tempo -= csm;
                                DC.k += k;
                                se DC.tempo <= 0 entao
                                        armazena antigeno, DC.k;
                                        renova DC;
                                        atualiza MCAV;
                                fim
                        fim
                fim
        fim
fim
para cada tipo de antigeno
        calcula metrica de anomalia Ka;
fim
```

Os antígenos analisados são representados pela variável antigeno, os sinais de entrada são representados pela variável sinal, a variável DC corresponde ao conjunto de células dendríticas, os sinais de saída derivados do processamento dos sinais de entrada correspondem às variáveis csm e k e MCAV é o grau de anomalia do antígeno. O MCAV pode-se considerado como uma variável de decisão e assim definir classificação de eventos de acordo com limiar padrão, um valor recomendado para este limiar é 0,65 em alguns casos conforme apresentado por Greensmith e Aickelin (2007).

O algoritmo possui basicamente as fases de iniciação, pré-processamento e processamento. Na fase de iniciação os parâmetros necessários são configurados, em seguida na fase de pré-processamento os valores dos sinais PAMPs, de perigo e seguros são coletados e é realizada

atualização destes valores e por fim é calculado o MCAV e a ação a ser executada na fase de processamento. A figura 2.10 apresenta as fases do algoritmo e a abstração para o ambiente computacional de rede.

Figura 2.10 – Fases do algorítimo de células dendríticas com abstração computacional de rede.



Fonte: Autor

2.5 Trabalhos Relacionados

Existem uma gama de trabalhos que utilizam como base o sistema imunológico humano para prover proteção aos sistemas computacionais. A teoria do perigo tratada por Matzinger (1994) apresentou uma abordagem que complementava a teoria do próprio e não próprio, pois garantia redução de falsos negativos, já que conhecimento do antígeno não era o única informação necessária. Com base nesta abordagem o algoritmo das células dendríticas foi apresentado por Greensmith, Aickelin e Cayzer (2005). Este algoritmo foi utilizado em Greensmith e Aickelin (2007) para comprovar sua eficácia para detecção de processos causadores de anomalias. Foi utilizado como antígenos um conjunto de processos definidos para comprovação dos resultados.

Anandita et al. (2015) refizeram o experimento tratado por Greensmith e Aickelin (2007) para comprovar a eficácia da teoria. No experimento constatou-se algumas limitações do DCA:

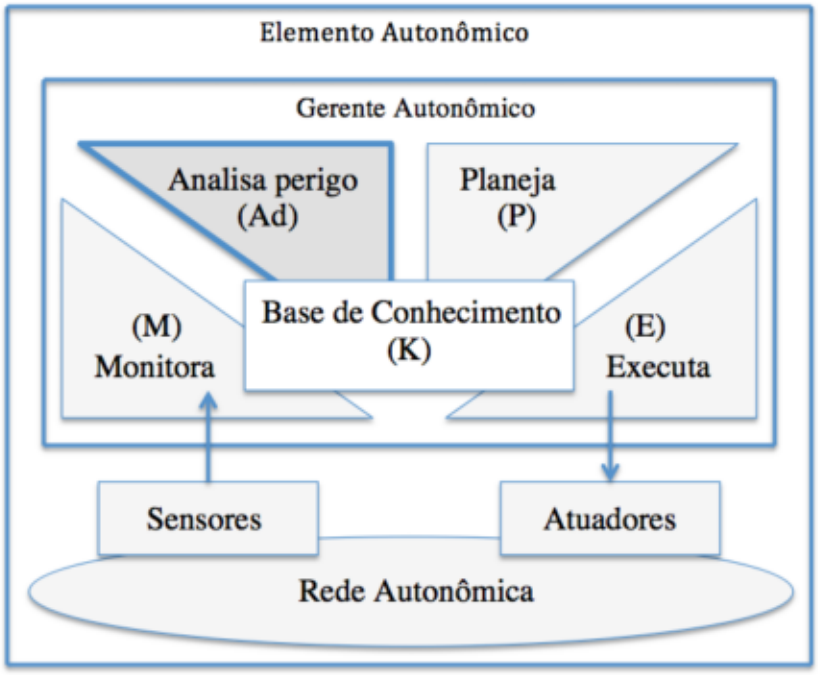
- O processamento de dados sobre células dendríticas não é realizado em tempo real de acordo com o processo de recuperação de dados de entrada.
- O processamento de dados é realizado diretamente no computador vítima, ele não usa a entrega de esquemas de dados.
- O antígeno foi coletado com base nos processos em execução no sistema relacionado ao identificador do processo PID (*Process ID*). Para cada sistema operacional, o modo de gerenciar os processo são distintos, dificultando a padronização do método de coleta independente do sistema operacional.

Vale ressaltar que ambos os experimentos apenas comprovam a eficácia do uso do algoritmo das células dendríticas para detecção de processos anômalos, porém não há reação contra a anomalia. Além de que, os experimentos utilizados nos trabalhos foram em feitos em um *host* com processos já definidos. Esta mesma abordagem foi utilizada por (AL-HAMMADI; AICKELIN; GREENSMITH, 2008) para detecção de um único *bot* em *host* comprometido com base nos atributos, apresentando bons resultados.

Já Oliveira, Salgueiro e Moreno (2013) apresentaram um modelo de alerta antecipado para detecção de anomalias, usando o algoritmo de células dendríticas determinísticas e simulando ambiente com proliferação de *malware confiker*. O *malware* executa o processo de varredura de porta (*Port Scan*) e este foi o foco do experimento que obteve resultados promissores. Outro ponto que merece destaque é a utilização do MAPE-K para definir o modelo proposto podendo expandir além da detecção. O modelo proposto por Oliveira, Salgueiro e Moreno (2013) foi denominado de MAdPE-K.O ciclo de gerenciamento e a arquitetura da gerência estão representados nas Figuras 2.11 e 2.12.

O sistema realiza automonitoramento através de mensagens de diagnóstico e histórico de informações, além de monitorar sensores do ambientes em sua volta. A partir destas informações, faz-se uma análise para a identificação de um possível evento anômalo. Sendo auxiliado por um conjunto de informações fornecidos por uma base de conhecimento.

Figura 2.11 – Arquitetura do ciclo MAdPE-K.



Fonte: (OLIVEIRA; SALGUEIRO; MORENO, 2013)

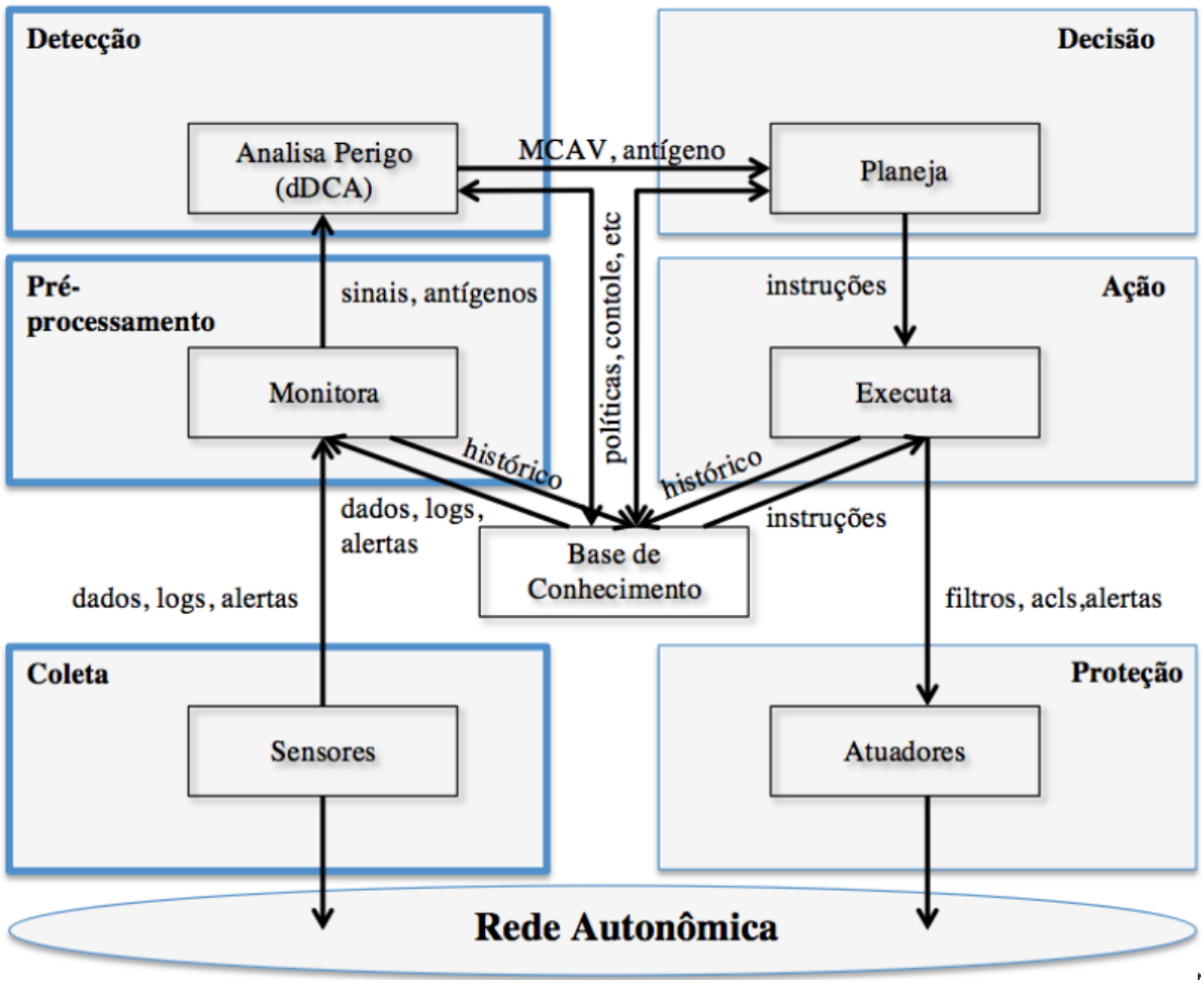
Neste modelo a existência de um gerente autonômico provendo as características autonômicas ao sistema e um ou mais elementos gerenciados. O ciclo de gerenciamento é definido em

quatro fases cada uma correspondendo a uma atividade do clico de gerenciamento autonômico, as fases são: pré-processamento, detecção, decisão e ação, e sendo todas auxiliadas por uma base de conhecimento.

Na fase de pré-processamento, os dados coletados pelos sensores ou armazenados na base de conhecimento são interpretados e transformados dados em informação. Já na fase de detecção as informações recebidas em forma de sinais e antígenos sendo usadas para identificar um contexto de perigo, é nesta fase onde é realizada a correlação de eventos, a detecção de anomalias e de seus possíveis causadores, tarefa realizada pelo o Algorítimo das Células Dendríticas Determinístico. A fase de decisão tem a responsabilidade de planejar as ações após recebimento dos resultados da fase de detecção e por fim na fase de ação. Esta fase é responsável por executar as instruções fornecidas pela fase de decisão (OLIVEIRA; SALGUEIRO; MORENO, 2013).

Todos os componentes, os fluxos dos dados e as fases da arquitetura, MAdPE-K estão representados na Figura 2.12.

Figura 2.12 – Arquitetura do modelo MAdPE-K.



Fonte: (OLIVEIRA; SALGUEIRO; MORENO, 2013)

Um outro modelo proposto tendo como base o modelo MAPE-K para combinar autono-

micidade e SDN foi proposta por Wendong et al. (2012), eles iniciaram um projeto para geração de um modelo autônomo para SDN baseado no protocolo *OpenFlow*.

- Camada física ou plano de dados: no modelo SDN autônomo baseado em *OpenFlow* que inclui *switches OpenFlow*, roteadores, outros elementos de rede.
- Camada de redes virtuais: os recursos físicos são virtualizados, pertencentes ao plano de dados, e consistem em vários tipos diferentes de redes virtuais afim de suportar serviços e aplicativos especificados. Assim esta camada é definida por um conjunto de recursos virtuais e programas e funções de controle e gerenciamento sendo executados por um controlador *OpenFlow*.
- Capa de controle de recursos de rede virtual: é responsável pela detecção, manutenção e alocação de recursos virtuais, e proporcionando o acesso programático lógico de controle de recursos de rede virtual à rede virtual.
- Camada de controle de serviço: quando os serviços específicos ou aplicativos são implantados em uma rede virtual, a lógica de controle de serviço dedicada seria construída automaticamente para controlar e gerenciar a implantação do serviço de acordo com seus requisitos específicos.
- Plano de Gerenciamento: Existem dois tipos de plano de gerenciamento o plano de gerenciamento de rede genérica e plano de gerenciamento de rede virtual.

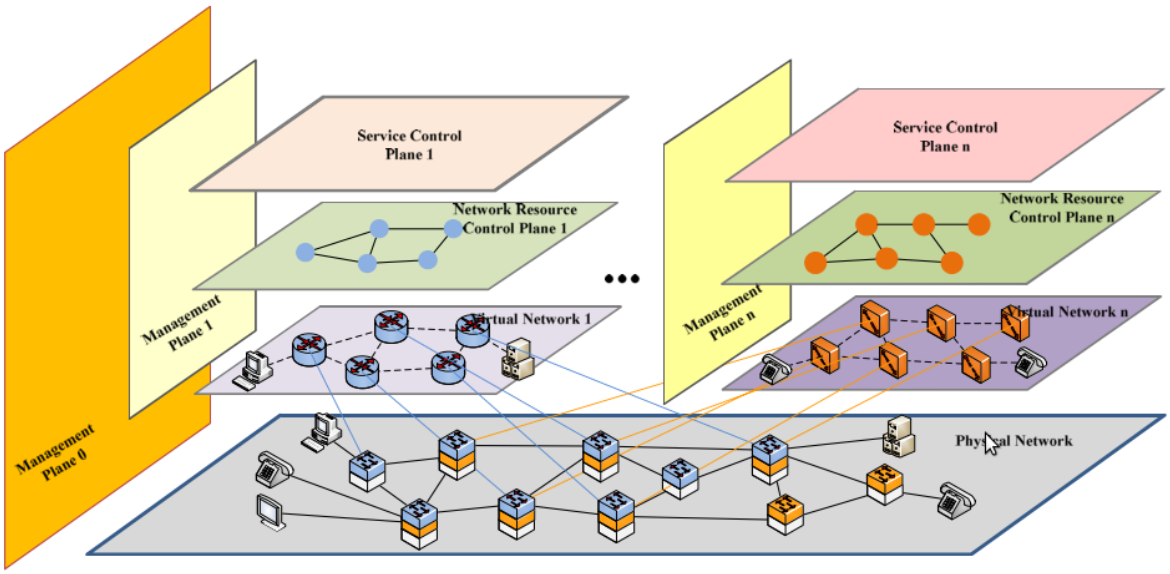
As 5 camadas do modelo estão ilustradas na Figura 2.13.

O modelo apresentado por Wendong et al. (2012) identifica a possibilidade de tornar a SDN autonômica, combinando os recursos deste paradigma e os requisitos para a autonomicidade.

As redes definidas por *software* provêm agilidade e dinamismo no controle e gestão da infraestrutura para o administrador. Devido às características da SDN, a possibilidade de mitigar ameaças existentes na rede tradicional tem sido amplamente pesquisada na academia, entretanto, também há riscos relacionados à própria estrutura da SDN e surgimentos de novas vulnerabilidades, assim ela pode ser usada como mecanismo de defesa para ataques conhecidos, mas faz-se necessário monitorar e analisar suas possíveis ameaças.

Em (YAN et al., 2015) é apresentado o estado da arte do uso das redes definidas por software e os possíveis ataques que possa sofrer. Os autores, apresentam as características da SDN que beneficiam a defesa cotra ataques de negação de serviço distribuído (DDoS) em computação em nuvem, as características pelos autores são: a separação dos planos de controle e dados, visualização e controle centralizado da rede, programabilidade da rede com uso de aplicações externas, análise de tráfego baseado em *software*, dinamicidade nas atualizações das

Figura 2.13 – Modelo SDN autonômico baseado em *Openflow*.



(WENDONG et al., 2012)

regras de encaminhamento e abstração dos fluxos de pacotes. Por outro lado, a SDN pode ser alvo de ataques DDoS, estes podem ser lançados contra as camadas de aplicação, controle e de dados.

Mihai-Gabriel e Patriciu (2014) apresentam uma maneira de mitigar ataques de negação de serviço (*Distributed Denial of Service - DDoS*) em redes definidas por software. A abordagem utilizada combina teoria do perigo e redes neurais para análise dos riscos. No experimento foi utilizado um sistema de gerenciamento e correlação de eventos de segurança (*Security Information and Event Management - SIEM*) conhecido no mercado, o AlienVault. A técnica utilizada no experimento para detecção do perigo foi feita inicialmente com uso de rede neural e posteriormente usando a abordagem da teoria do perigo do próprio e não próprio para detecção do risco, conforme apresentado por Matzinger (1994). O experimento teve resultado satisfatório para mitigar um ataque DDoS em tempo médio de 24 segundos, porém a reação não foi feita utilizando agentes autonômicos, apesar de evidenciar a possibilidade no modelo proposto para detecção de anomalias o trabalho não utilizada de elementos autonômicos, e sim das características de programabilidade das redes definidas por software para mitigar o ataque.

2.6 Resumo

Neste capítulo foram tratados os conceitos fundamentais sobre segurança de redes e evidenciando os ataques de varredura de portas, ataque utilizado para o caso de uso utilizado neste trabalho. Conceitos redes definidas por software, tratando de sua arquitetura e características que permitem mudaram a maneira de gerenciar e administrar ambientes computacionais além

de suas fragilidades. Neste capítulo foram apresentado os fundamentos e os elementos de uma rede autonômica. Outro conceito fundamental para este trabalho e apresentado neste capítulo fo a abordagem da teoria do perigo, utilizada para detectar eventos anômalos. E por fim os trabalhos relacionados que de alguma forma abordaram os conceitos expostos com intuito de prover segurança e/ou autonomicidade em redes computacionais.

3

Serviço MAdPE-K/SDN

Inspirado pelo modelo de gerência autonômico com foco em segurança, MAdPE-K, e pelos resultados obtidos por Oliveira, Salgueiro e Moreno (2013) na predição de ataques, implementou-se o serviço denominado MAdPE-K/SDN (*Software-Defined Network with MAdPE-K*) para mitigar ataques em uma rede SDN. O elementos que contidos na implementação do MAdPE-K/SDN está distribuída nas camadas da SDN e provê autoproteção e autogerenciamento, tendo em vista as características de programabilidade, gestão centralizada da SDN, este serviço está apresentado na 3.2.

Na arquitetura do serviço MAdPE-K/SDN, a interação entre as camadas SDN e a arquitetura MAdPE-K é realizada na camada de infraestrutura durante a fase de monitoramento. E na camada de aplicação durante a fase de execução da ação mitigadora com utilização de APIs.

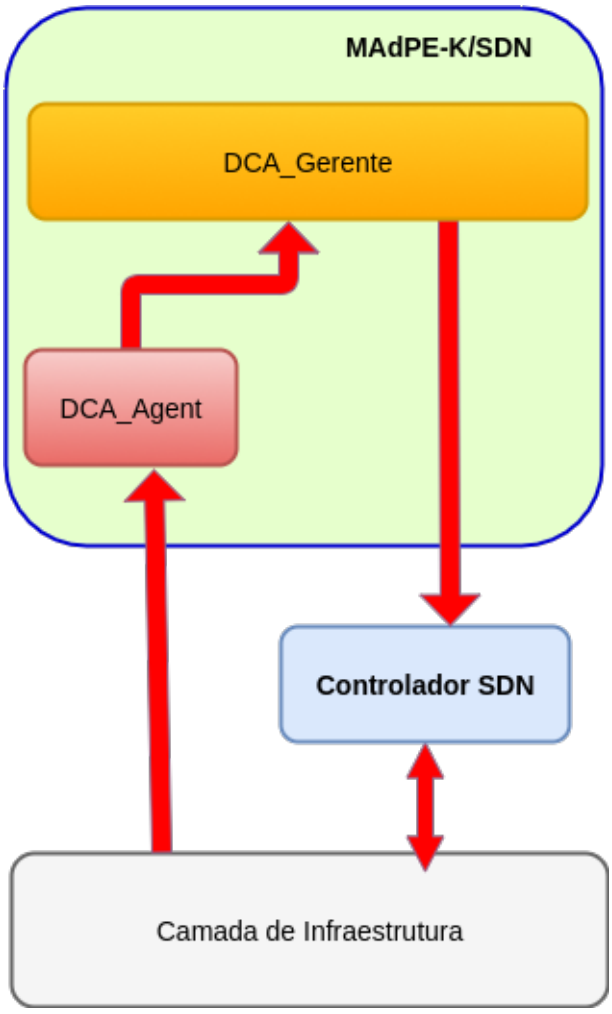
A base de conhecimento tem a responsabilidade das trocas de informações entre todas as fases, auxiliando-as no desempenho de suas funções. Sendo responsável pelo autoconhecimento devido às informações que gerencia que podem ser de controle, de políticas, histórico, instruções, dados, logs, alertas,entre outros.

Em suma, o processo de monitoramento interage com a camada de de infraestrutura através dos agentes, em seguida é direcionado para a fase de análise do perigo que, em caso de detecção de evento anômalo, e tendo como base as estratégias definidas no planejamento a fase de execução entra em ação interagindo com a camada de aplicação e está comunica-se com a a camada de controle enviando ao controlador as mensagens contendo a ação que teve realizar para mitigar o evento detectado.

A Figura 3.2 apresenta a interação entre arquitetura SDN e o modelo MAdPE-K.

As fases da arquitetura do serviço MAdPE-K/SDN estão detalhadas nas seções que se seguem da seguinte maneira: na seção 3.1 monitoramento, na seção 3.2 a fase de análise do perigo, na seção 3.3 a fase de planejamento e na seção 3.4 a fase de execução.

Figura 3.1 – Serviço MAdPE-K/SDN



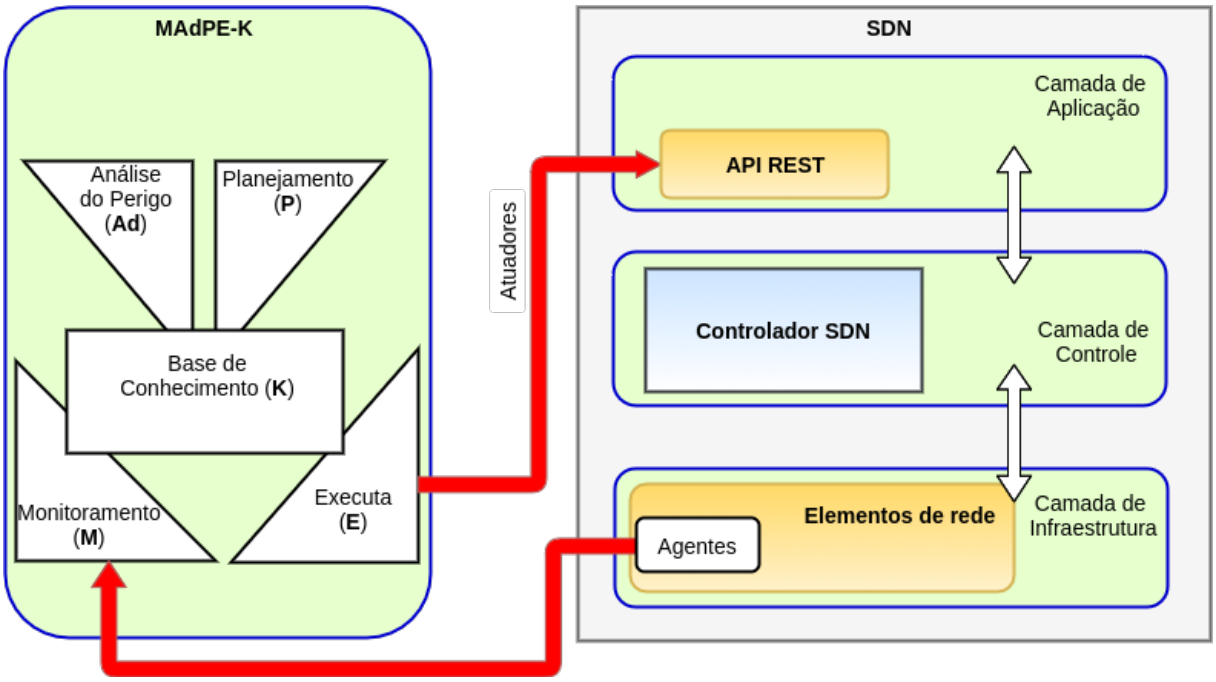
Fonte: Autor

3.1 Monitoramento

A fase de monitoramento, apresentada na Figura 3.3, compreende a coleta dos dados dos dispositivos que integram a rede. Estes dados podem ser obtidos de arquivos de *Logs*, alertas ou dados fornecidos pelo sistema operacional. A coleta é realizada por sensores que agem diretamente na camada de abstração de dispositivos e recursos ou utilizado o serviço de monitoramento da SDN, ambos através da interface *Southbound*.

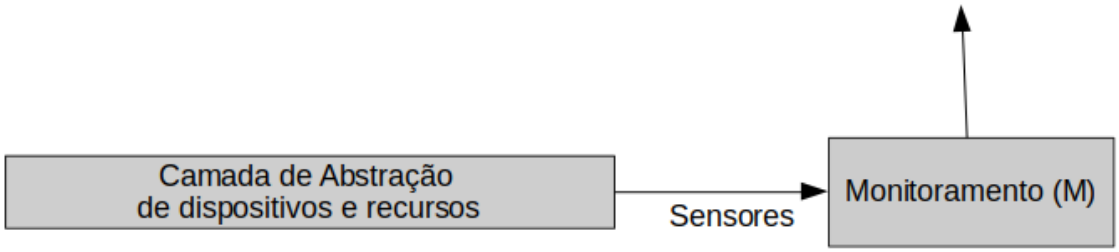
A localização e a quantidade dos sensores afetará de modo significativo nos resultados da coleta. Os sensores agem diretamente nos dispositivos coletando dados específicos do equipamento e devem estar presentes em todos os dispositivos para uma eficiente coleta dos dados. Os dados que estão relacionados ao fluxo e serviços da SDN podem ser coletados utilizando os serviços existentes na estrutura, como logs, fluxo de dados e status da rede.

Figura 3.2 – Arquitetura do serviço MAdPE-K/SDN



Fonte: Autor

Figura 3.3 – Elementos e fluxos da fase de coleta



Fonte: Autor

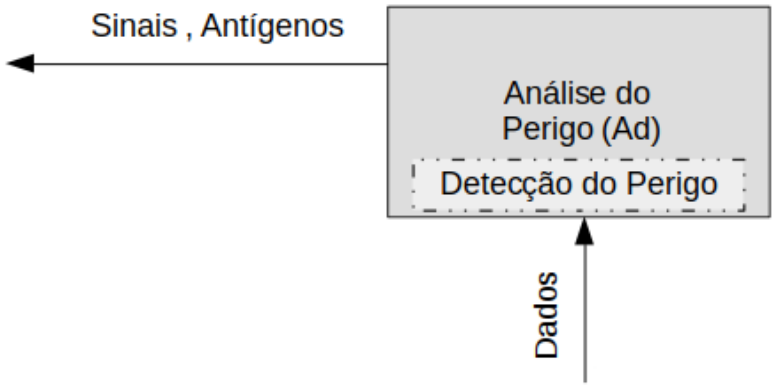
3.2 Análise do Perigo

A fase de análise do perigo compreende o recebimento e processamento dos dados para a geração de sinais (PAMP, sinal seguro e sinal de perigo). Esta fase é realizada pelo algoritmo das células dendríticas determinístico (dDCA), sendo responsável pela detecção antecipada do perigo.

A ocorrência de uma potencial anomalia é resultado da análise executada nesta fase. O envio de sinais e antígenos para a fase de planejamento só ocorrerá em situação de perigo.

A fase análise do perigo esta representada na Figura 3.4.

Figura 3.4 – Elementos e fluxos da fase de Análise do Perigo



Fonte: Autor

3.3 Planejamento

Nesta fase é que as ações são planejadas as instruções para mitigar os ataques. Ao receber os sinais e antígenos, após a fase de análise onde há detecção do perigo, são verificadas as políticas definidas pela gestão de rede. É importante ter em mente que a política é definida pela gestão para atender aos requisitos necessários para manter a disponibilidade e confiabilidade dos serviços e recursos na rede.

O fato da SDN ter uma gestão e monitoramento centralizados permite políticas de rede em um alto nível de abstração, possibilitando a criação de políticas para tomada de decisão com base nos sinais de perigo recebidos na fase de planejamento.

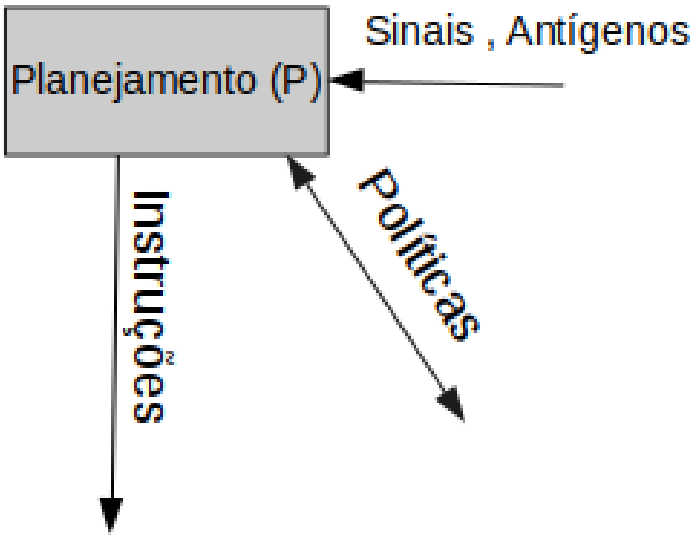
A Figura 3.5 representa os elementos que compreendem esta fase.

3.4 Execução

A fase de execução, representada na 3.6, tem a responsabilidade de executar as instruções recebidas da fase de planejamento. A execução é realizada pelos atuadores, componentes que efetivam as ações de mudança no ambiente.

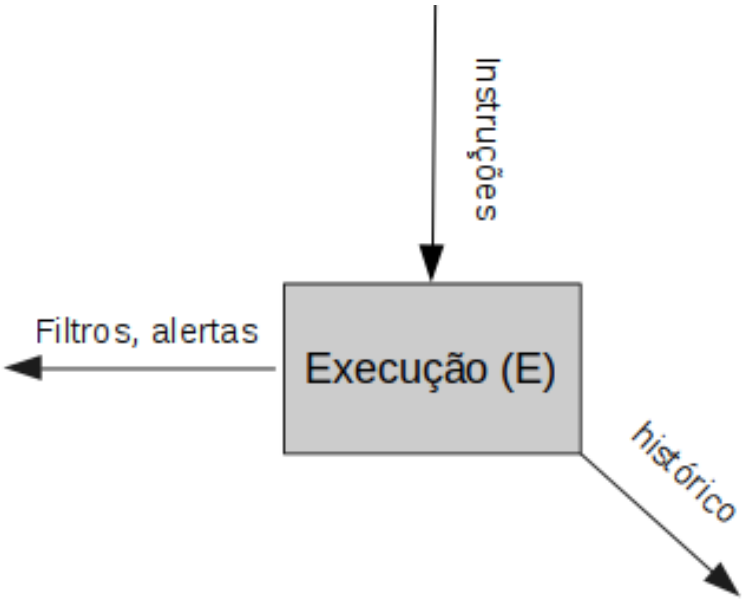
Em redes definidas por software os atuadores serão as APIs, as ações são efetivadas com envio ds mensagens ao controlador através da interface (*Northbound*). Os formatos das mensagens contendo as ações dependem de quais modelos de mensagens são suportadas pelo controlador SDN em questão.

Figura 3.5 – Elementos e fluxos da fase de Planejamento



Fonte: Autor

Figura 3.6 – Elementos e fluxos da fase de Execução



Fonte: Autor

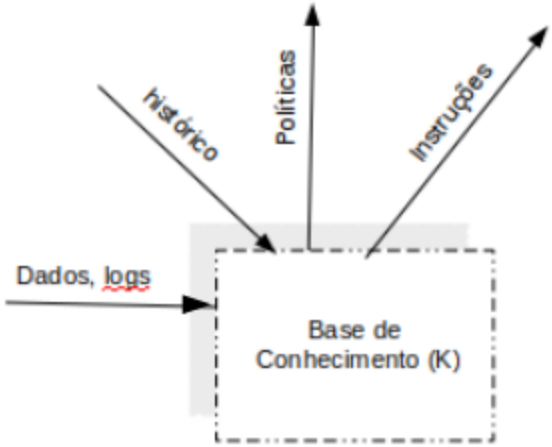
3.5 Base de conhecimento

A base conhecimento tem a responsabilidade de receber e fornecer informações para as outras fases de monitoramento, planejamento, análise do perigo e execução, auxiliando nas atividades de cada uma das fases.

As informações referentes ao *status* da topologia rede e do fluxo são obtidas da interface de comunicação (*Northbound*) da SDN, devido a características de centralização do SDN. Já as informações referentes aos *status* dos dispositivos através da camada de abstração de dispositivo e de recursos (*Southbound*) podendo também serem coletadas diretamente dos arquivos de logs do sistema dos dispositivos.

A Figura 3.7 apresenta o elemento da base de conhecimento e como interage com os outros elementos. Recebe como entradas histórico de fluxo e status da rede, dados e logs enviado dos sensores localizados nos nós da rede e fornece políticas para a fase de planejamento e instruções para a fase de ação para tomada de decisão.

Figura 3.7 – Elementos e fluxos da base de conhecimento



Fonte: Autor

3.6 Implementação do Serviço MAdPE-K/SDN

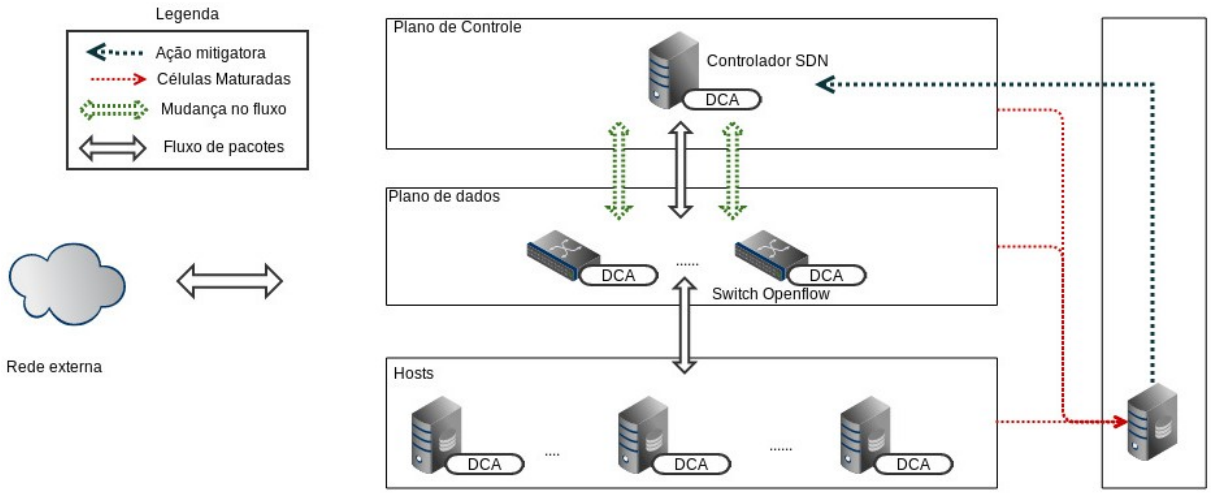
Neste trabalho aplicamos as ideias em cenário real para garantir sua aplicabilidade num contexto geral, além de considerar o elemento central da reação os sinais de perigo que foram coletados a partir de dados referentes ao comportamento do nó e desconhecendo os processos em execução, garantindo assim a detecção de sinais de alerta com base o comportamento do nó. Esta abordagem vai de encontro à que foram utilizadas em outros trabalhos em que a detecção teve como objetivo único identificar quais dos processos gerava sinais de alerta e assim identificando qual deles eram anômalos.

As fases de monitoramento e processamento é realizado pelo módulo *dca_agente*. Nesta etapa é realizada as coletas dos dados extraídos diretamente do dispositivos e processados para definição dos sinais e análise do perigo, gerando as estruturas de células dendríticas que serão processadas na a fase de análise do perigo.

As fases de análise do perigo e execução são de responsabilidade do módulo *dca_gerente*, este módulo recebe as estruturas de células e verifica se há uma possível anomalia comparando as

quantidades de células dendríticas maturadas em relação às semimaduras. Caso detecte anomalia, envia uma mensagem para ativar a resposta imune recebida pelo servidor REST (*Representational State Transfer*) do controlador através de API usando arquivos no formato JSON (*JavaScript Object Notation*).

Figura 3.8 – Sistema MAdPE-K/SDN



Fonte: Autor

A Figura 3.8 representa o ambiente proposto. Cada ativo contém um agente (sensor) que coleta e organiza os dados. Estes agentes ainda têm a responsabilidade de processar os dados coletados com a utilização do algoritmo de células dendríticas para análise do perigo. Em caso de perigo, uma saída contendo as células migradas é enviada ao *dca_gerente*, servidor responsável por receber os dados. Assim, se a quantidade de células migradas com status de maturadas forem maiores que semi-maturadas, é enviado ao controlador SDN instrução para prover a reação ao ataque, como por exemplo, filtragem de pacotes, monitoramento, isolamento dos ativos, etc.

Com o propósito de permitir melhor visualização e especificidade da proposta foram criado três diagramas: *observer*, de atividades e de implementação.

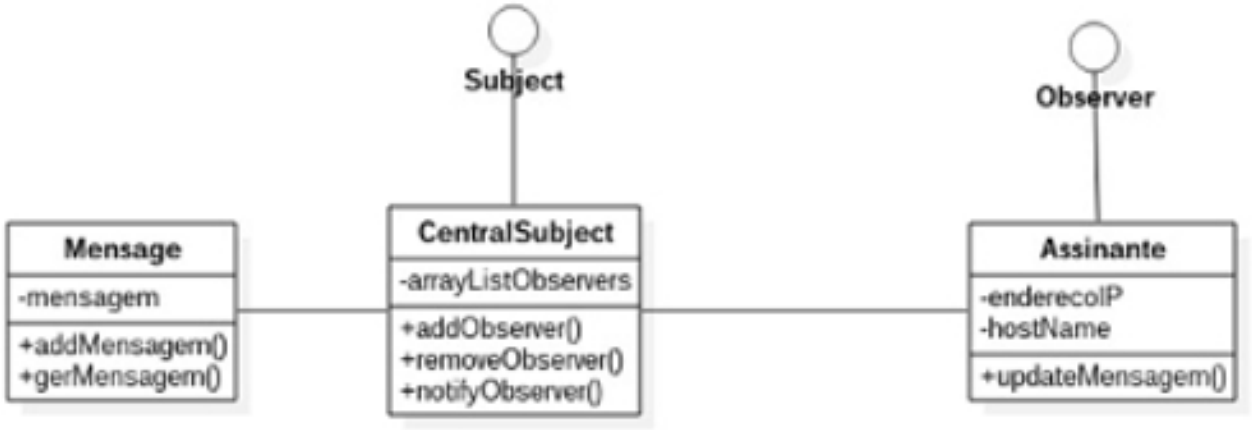
O diagrama representado pela Figura 3.9 é um dos 23 padrões GOF (*Gang of Four*) que representa a mensagem contendo todos os atributos que devem ser comunicados (LARMAN, 2002). O padrão *Observer* escolhido para esta representação provê uma forma de notificar eventos em uma dependência de um para muitos entre os objetos *Mensage*, pois permite o desacoplamento das classes, melhorando a comunicação entre os componentes de forma padronizada e atestada por possuir boa documentação.

O objeto *Mensage* contem os dados recebidos pelo sistema de células dendríticas, de forma que quando o estado de um deles é alterado, um aviso é enviado aos objetos observadores (ENGHOLM, 2013). Já o objeto *CentralSubject* é o responsável por definir quem vai receber a mensagem, o destino da mensagem é definido no objeto *Assinante*.

A modelagem do sistema proposto esta representada pelo diagrama de implementação na Figura 3.10, mostrando o relacionamento entre os componentes de hardware e software no sistema tendo como foco as configurações dos componentes.

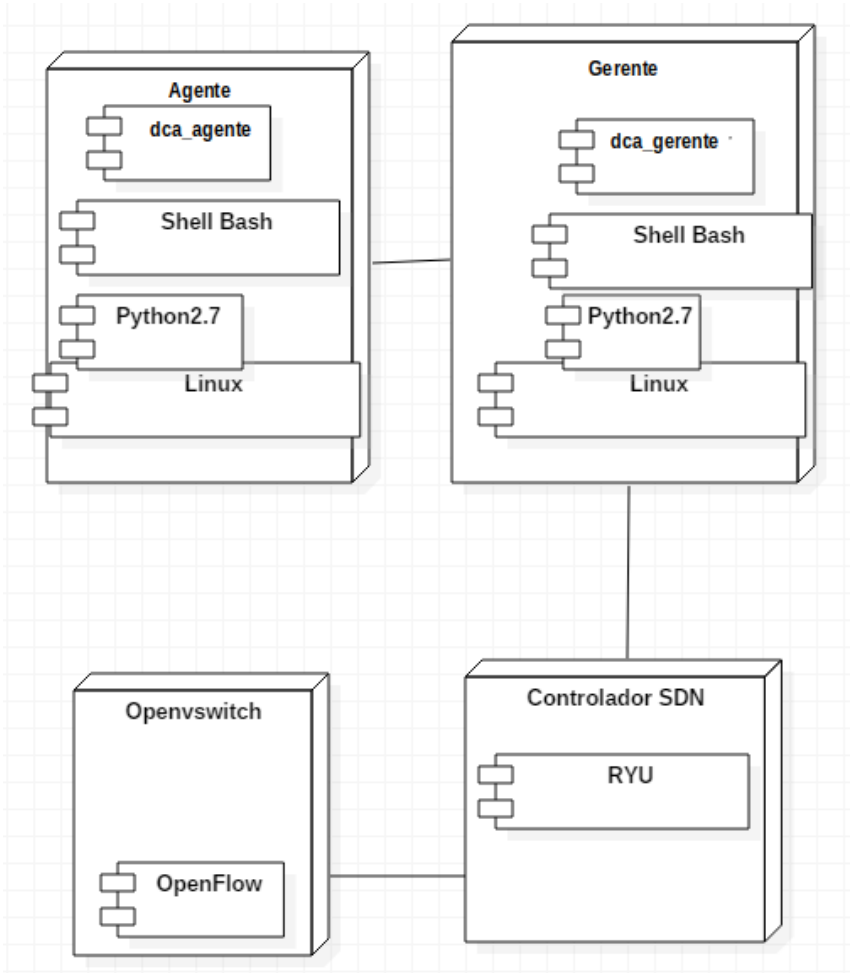
O diagrama de atividades representado na Figura 3.11 mostra as sequência de atividades do processo (LARMAN, 2002). O cliente, nó possivelmente infectado, utiliza o *dca_agente* para analisar os dados coletados e gerar os sinais PAMP, seguro e de perigo para, após processamento ser encaminhando para o servidor responsável pela gerencia, fazendo assim o papel do linfócito. No servidor, o dca_gerente analisa e sinaliza o controlador SDN para uma possível reação.

Figura 3.9 – Diagrama Observer para MAdPE-K/SDN.



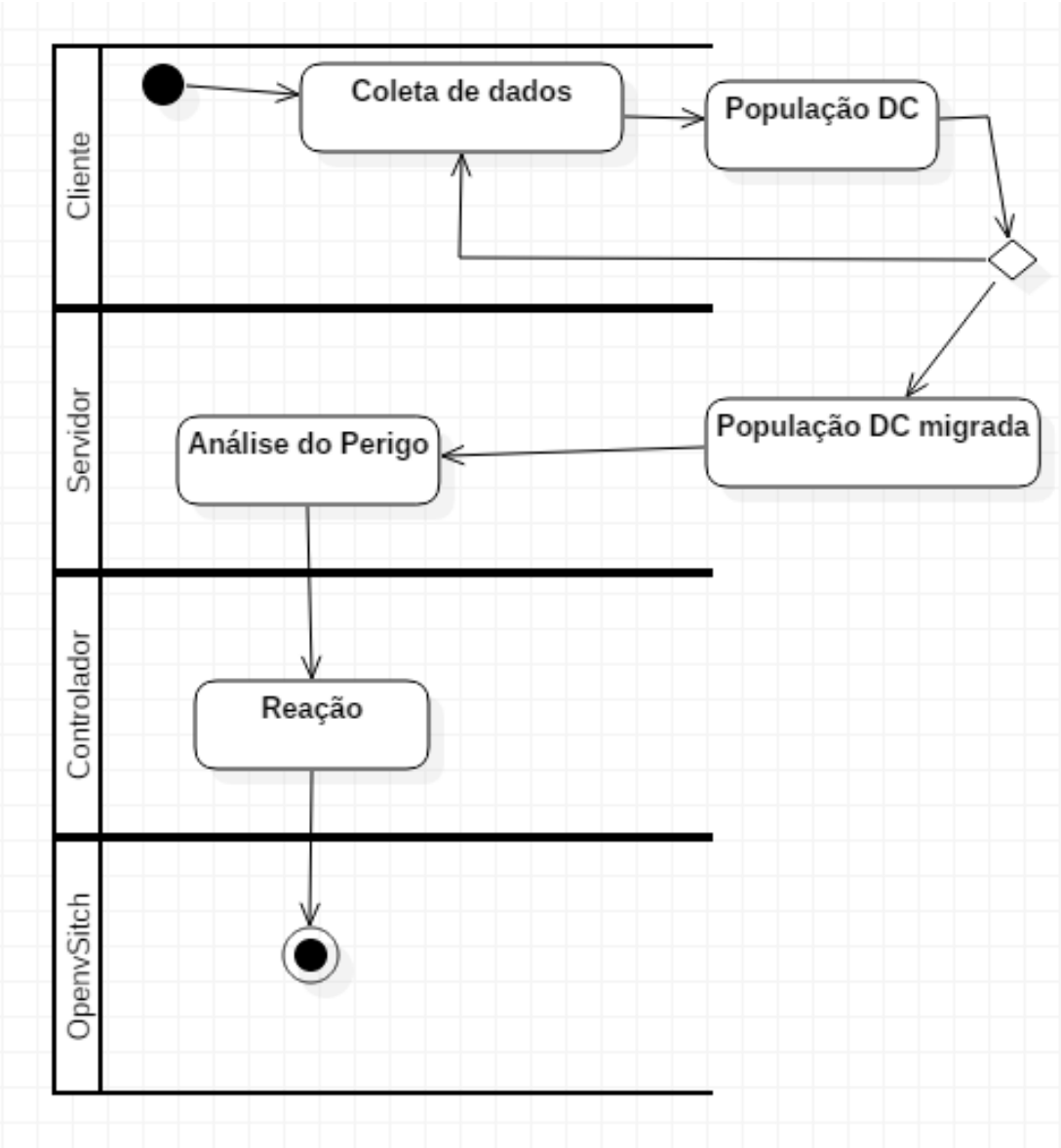
Fonte: Autor

Figura 3.10 – Diagrama de implantação para MAdPE-K/SDN



Fonte: Autor

Figura 3.11 – Diagrama de Atividades para MAdPE-K/SDN



Fonte: Autor

4

Caso de Uso

Com o intuito de analisar a viabilidade do serviço MAdPE-K/SDN, foi desenvolvido um sistema onde foram implementadas todas as fases do modelo MAdPE-K conforme descrito po Oliveira, Salgueiro e Moreno (2013). Nas seções que seguem estão descritas o ambiente de testes e as fases do ciclo MAdPE-K para proteção de um ataque de *Ping Scan*, como estudo de caso.

4.1 Experimento

O experimento teve como base os os casos de uso tratados por (ANANDITA et al., 2015), (GREENSMITH; AICKELIN; TWYCROSS, 2006) e (OLIVEIRA; SALGUEIRO; MORENO, 2013), a detecção de ataque de reconhecimento com uso do padrão de ataques do tipo *Ping Scan*, esta técnica de ataque é utilizada nas primeiras etapas de um ataque em buscar identificar os ativos e os serviços em execução em um segmento de rede, desse modo possibilitando melhores chances de sucesso em ataques de negação de serviço, de roubo de dados, escalação de privilégios ou disseminação por *malware*.

Há inúmeras ferramentas, *malwares* e técnicas de varredura que podem ser utilizadas por atacantes. Além de que, em uma infraestrutura real, pode haver uma grande quantidade de ativos que podem distinguirem-se em *hardware* e sistemas operacionais. Com isso em mente, neste trabalho tratamos como sinais de perigo o comportamento do conjunto de processos de um ativo, ao invés de considerar cada processo sendo um antígeno como tratado por Greensmith e Aickelin (2007). Assim, neste trabalho, a análise foi desenvolvida com base no comportamento anômalo independente do conhecimento dos outros processos que estejam em execução.

A Tabela 4.1 ilustra a correlação entre os sinais do sistema imunológico humano e o sistema de comunicação.

Tabela 4.1 – Correlação entre os sistemas imunológico humano e segurança computacional

Sistema Imunológico Humano	**Sistemas computacionais**
Atividade externa anomala	Ataque de varredura de ativos e servços abertos
Atividade externa normal	Atividade esperada
Sinais Pamp	O valor de erros ICMP de destino inalcansável por segundo
Sinal de Perigo	A quantidade de pacotes de dados transmitidos para a rede externa.
Sinal Salvo	Inversão da taxa de alteração do pacote de dados que é enviado.
Antígeno	Total de chamadas de sistema do processo ativo no sistema de computador.
Maturação de células dendriticas	Uma condição em que o valor de sinal maduro é maior do que o valor do sinal semimaduro. Indicado pelo valor do contexto de célula que é 1
Semimaturação de céluas dendriticas	Uma condição em que o valor de sinal semimatura é maior do que o valor do sinal maduro, é indicado pelo valor do contexto de célula que é 0.
Sinal de Perigo de uma célula maturada	Coeficiente de Valor Antigénio de Contexto Maduro - Coefficient of Mature Context Antigen Value (MCAV).

Fonte: (ANANDITA et al., 2015)

4.1.1 Ambiente de testes

Nesta dissertação foi utilizado emulador que, ao contrário dos simuladores, imita o comportamento de um conjunto de hardware de rede ao invés do comportamento descrito através de escalonamento de eventos. Logo o cenário para a análise do sistema proposto foi baseado em ambientes reais, ou seja, sem uso de simuladores.

Os testes foram feitos com o objetivo de ser o mais fiel possível a um ambiente de redes real, todos os sistemas operacionais e serviço foram configurados individualmente com uso emulador de redes, assim verificando a aplicabilidade em redes operacionais.

Para os experimentos foi utilizada a abordagem do Algoritmo de células dendríticas determinísticas *dDCA* (GREENSMITH; AICKELIN; TWYCROSS, 2006) e dos sinais coletados de um ataque do tipo *PORT SCAN*, apresentados em trabalhos similares que comprovaram a eficiência na coleta e determinação de anomalias (OLIVEIRA; SALGUEIRO; MORENO, 2013) (ANANDITA et al., 2015).

4.1.2 Recursos utilizados no experimento

Os recursos computacionais utilizados para o experimento foram:
Infraestrutura Base:

- Computador pessoal ou *laptop* com as seguintes especificações:
  - Processador: Intel®Core(TM) i7-3632QM CPU @ 2.20GHz
  - RAM: 8GB
  - Sistema Operacional: Ubuntu Linux 16.04 LTS

Cenário:

- Virtualização: VirtualBox;
- Comutação de pacotes: Contêiner Docker do OpenvSwitch 2.5;
- Sistemas Operacionais Virtualizados: VirtualBox 16.04 LTS;
- Emulador: GNS3;
- Controlador RYU.

Linguagens de programação utilizadas:

- Scripts em Bash para coleta dos dados para formação dos sinais
- Linguagem de programação Python para implementação da comunicação entre o gerente e o controlador RYU.

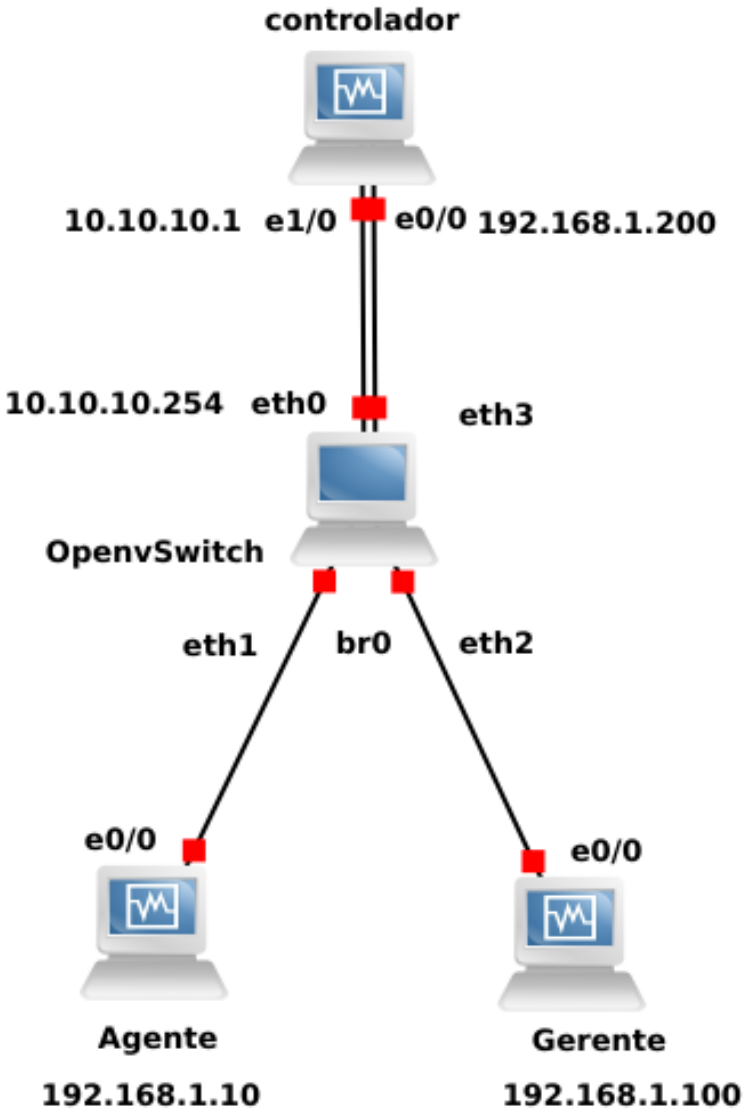
Um ambiente emulado foi montado utilizando o GNS3 e composto de três máquinas virtuais e um contêiner *docker* gerenciado. Este cenário pode ser expandido e a possibilidade de haver comunicação com um hardware externo, além do fato de os *switchs* em contêiner facilitarem a implantação de uma quantidade maior de elementos sem excessivos consumos de recursos.

A Figura 4.1 apresenta a topologia criada para execução dos experimentos. As descrições das máquinas virtuais e configurações de sistemas dos ativos Agente, Gerente e controlador são idênticas, ambiente virtualizado com uso do VirtualBox integrado ao GNS3 e sistema operacional Ubuntu 16.04 (64 bits) e memória RAM de 512 MB. O contêiner OpenvSwitch pode ser encontrado no repositório oficial do GNS3.

A comunicação entre os equipamentos através das interfaces do OpenvSwitch com duas redes distintas. As interfaces eth1, eth2 e eth3 do elemento central da rede estão inseridos na *br0* e os elementos a eles conectados formam a rede gerenciada pelo controlador. A interface eth0 do

OpenVswitch e a interface e1/0 do controlador foram configuradas para rede de gerenciamento, ou seja, caso necessite intervenção direta entre o controlador e o OpenvSwitch em caso de uma reconfiguração onde a rede SDN esteja inoperante.

Figura 4.1 – Topologia Usada no Experimento



Fonte: Autor

4.2 Análise das fases MAdPE-K/SDN

Nas seções seguintes estão descritas a aplicação de todo o ciclo do modelo MAdPE-K na arquitetura proposta como, também, os resultados do experimento.

4.2.1 Monitoramento

A coleta teve foco nos dados que remetem a uma investida de varredura de um nó interno da rede em busca de colher informações com uso da técnica SYN SCAN, assim os dados foram coletados das informações do *host* sem adição de nenhuma ferramentas de análise de pacotes (*sniffer*) externa.

Conforme abordado por (OLIVEIRA; SALGUEIRO; MORENO, 2013), (ANANDITA et al., 2015) e (GREENSMITH; AICKELIN, 2007), na fase de coleta dos dados para processamento de sinais de entrada os dados foram coletados com intervalos de 1 (um) segundo e foram obtidos utilizando *script* em Bash e os arquivos /proc/net/snmp e /proc/net/dev, arquivos que contém estatísticas das comunicações e das interfaces de rede respectivamente. O sinal PAMP é proveniente da quantidade de pacotes do protocolo ICMP com erros, que são obtidos no arquivo /proc/net/snmp. Os sinal de perigo foi definido como a quantidade de pacotes enviados pelo *host* e obtidos no arquivo /proc/net/dev. Os dados para a geração do sinal seguro foi definido com base à derivação do sinal PAMP gerado.

4.2.2 Análise do Perigo

O processo de detecção utiliza o algorítimo de células dendríticas que processa os sinais do *host* para prover o modelo de sistema de alerta antecipado (*EWS - Early Warning System*) não hierárquica e distribuído (OLIVEIRA; SALGUEIRO; MORENO, 2013) e alerta o agente para tomada de decisão. O mapeamento dos sinais tiveram como base a Tabela 4.1 sendo assim definidos:

- **Sinal de PAMP1**: número de pacotes ICMP com mensagens de erro *"destination unreachable"* recebidas por segundo.
- **Sinal de PAMP2**: sinal de perigo enviado por outro host, sendo 0 quando não houver alertas de perigo e valor 1 indicando alerta de anomalia.
- **Sinal de Perigo (SP)**: número de pacotes TCP enviado em relação ao número total de pacotes.
- **Sinal Seguro (SS)**: desvio padrão dos sinais PAMPs.

O valor de limiar utilizado para MCAV foi de 0,5 conforme resultados do experimento feito por (ANANDITA et al., 2015) tendo em vista que os sistemas operacionais e os softwares e técnicas de escaneamento utilizadas nos experimentos aqui relatados são semelhantes.

Os pesos utilizados para o processamento do sinais foram baseados na sugestão de Greensmith, Aickelin e Cayzer (2005) e apresentados na Tabela 4.2.

Tabela 4.2 – Pesos Usados

W	CSM	semi	mat
PAMPs(P)	2	0	2
Sinal de Perigo	1	0	1
Sinal Seguro	2	3	-3

Após a coleta na fase de monitoramento, na fase de análise do perigo os dados são processados e são feitos os cálculos das potências dos sinais através da equação 2.1.

A Figura 4.2 apresenta os valores dos sinais na ocorrência de um teste de conectividade entre dois ativos usando o protocolo *ICMP* sem perdas. Percebe-se mais incidência os valores referentes aos sinais seguro que o sinal PAMP2 e valores próximos ao PAMP1. Tendo como base a utilização dos pesos da tabela 2.2 os sinais seguros suprimi os outros sinais acarretando em uma situação normal, evitando assim a ocorrência de falsos positivos.

Já na Figura 4.3 foi coletado dados considerando falhas nos testes de conectividade entre ativos, situação comum considerando ativos inoperantes ou problemas de comunicação. Fica evidente um valor maior do sinal PAMP1 e do sinal de perigo em relação ao gráfico 4.2, entretanto não o suficiente para ativar uma reação.

Por fim, a Figura 4.4 representa as potências dos sinais dos dados coletados durante a execução de um ataque de escaneamento com uso do *SYN SCAN*. Os valores das potências dos sinais de perigo e PAMP1 são muito superiores ao sinal seguro, evidenciando assim um situação de perigo.

Vale ressaltar que os valores do sinal PAMP2, neste experimento, foi definido como 0, pois foi considerado que não há sinal de perigo.

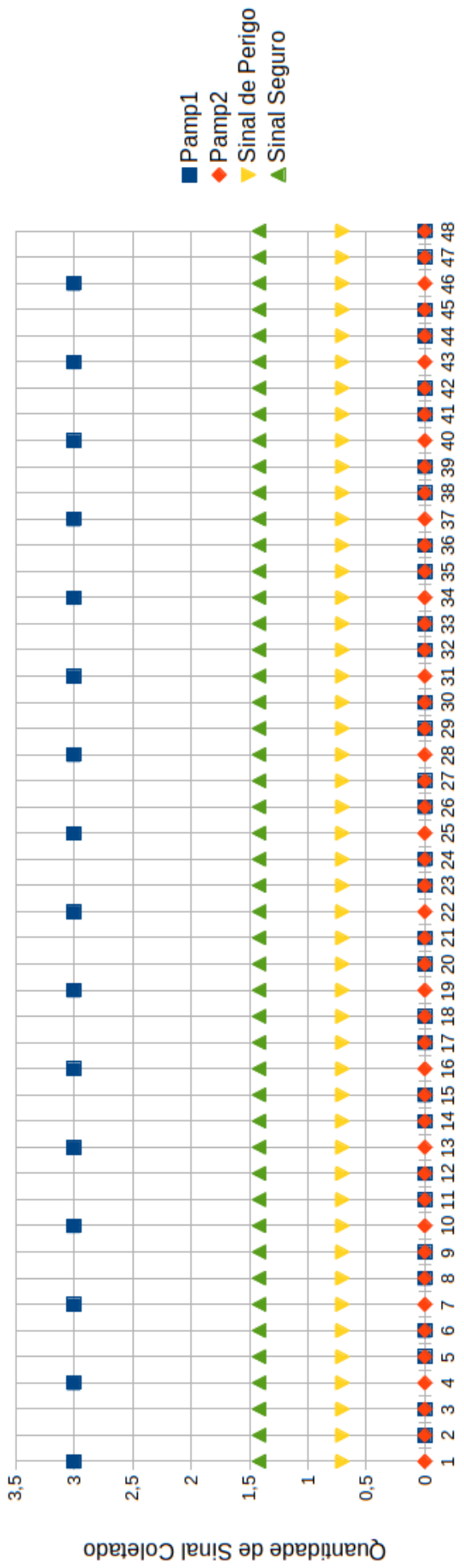


Figura 4.2 – Sinais Coletados em um fluxo sem anomalias sem falhas de ICMP.

Fonte: Autor

Figura 4.3 – Sinais Coletados em um fluxo sem anomalias com erros de ICMP

Fonte: Autor

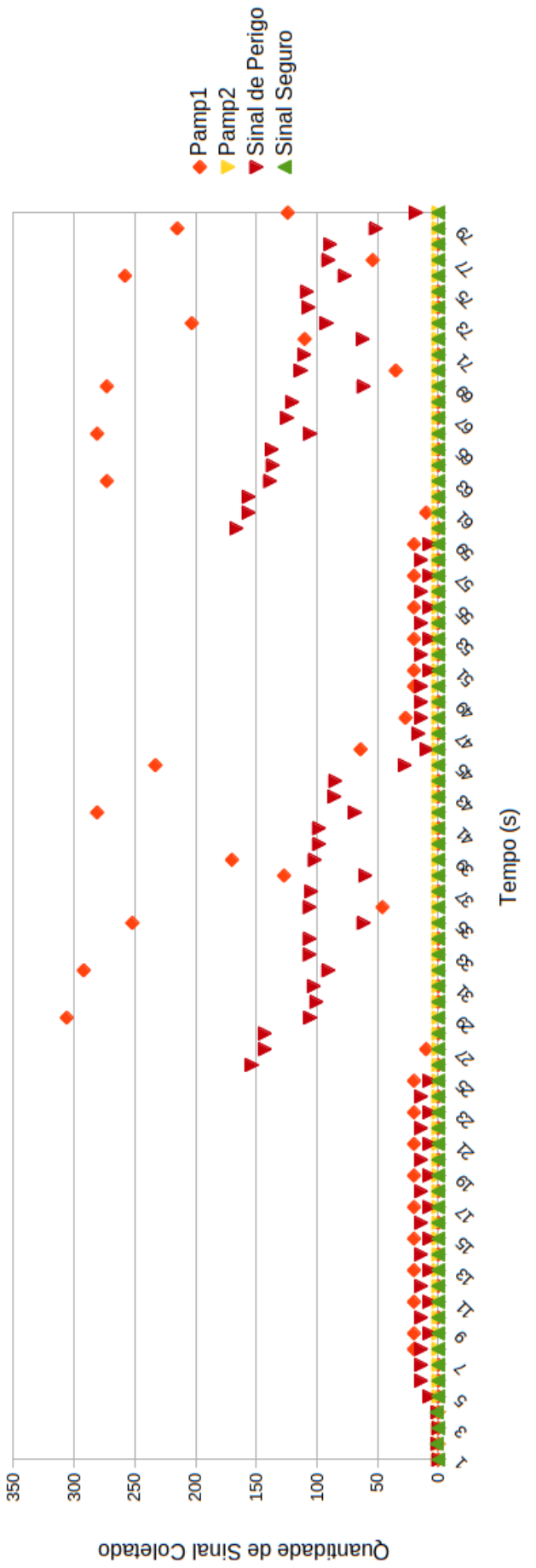


Figura 4.4 – Sinais Coletados em um fluxo com anomalias (SYN SCAN)

Fonte: Autor

4.2.3 Planejamento e Execução

Na ocorrência da detecção do perigo, o *dc_agent* envia uma mensagem para o servidor REST no controlador, alterando o fluxo no *Open vSwitch*, isolando o *host* da rede afim de proteger a infraestrutura e mitigar o ataque.

Um exemplo de mensagem está ilustrado em no código 4.1. O servidor REST utilizado foi o padrão existente no próprio controlador RYU, sem nenhuma alteração. A mensagem contém informações para controle do fluxo, que neste caso, adiciona um nova regra de fluxo para bloquear as comunicações que se destinam ao *host* que apresenta anomalia. Os parâmetros utilizados para formação da mensagem foram:

- *priority* = prioridade da nova entrada.
- *dpid* = identificação do *switch*.
- *dl_dst* = endereço físico da interface do *host* que apresentou anomalia.
- *actions* = ação é bloquear a comunicação na interface 1.

Código 4.1 – Mensagem JSON enviada ao controlador

```
{

"dpid": "222122511827781",

"priority": 1000,

"match": {

        "dl_dst": "08:00:27:0a:a9:ba"

        },
"actions":[

        {
                "type":"DROP",
                "port":1
        }
        ]
}
```

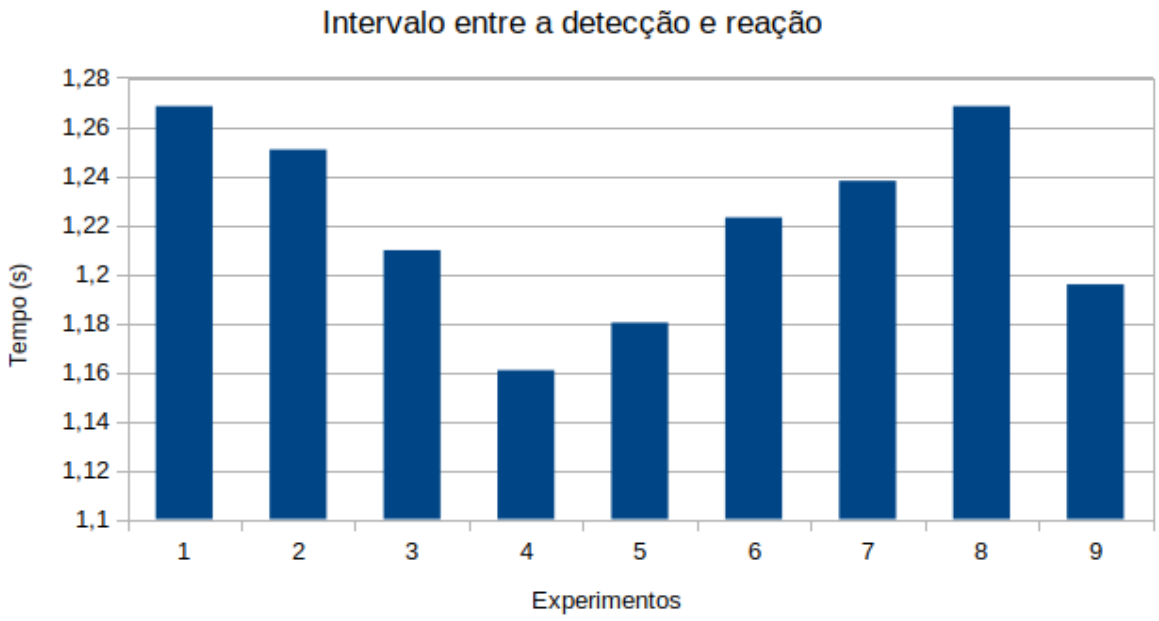
A mensagem enviada ao controlador pelo código 4.1 possui os parâmetros mínimos para a criação de uma nova entrada na tabela de fluxos que deve ser aplicada ao *OpenvSwitch* ao qual o vetor de ataques está conectado. Outras entradas poderiam ser criadas ou modificadas a

depender da necessidade do administrador e da política de segurança e de tratamento a incidentes aplicada a rede.

A reação teve um tempo de resposta em média de 1,216 segundos para o isolamento do *host* de onde se originou o ataque, ação definida para reação a este tipo de ataque. Importante ressaltar que os valores do tempo de resposta foram obtidos com 99% de confiança resultando em um intervalo de [1,17:1,26].

Os tempos de reação dos experimentos estão ilustrados na Figura 4.5.

Figura 4.5 – Intervalo entre o processamento e a reação.



Fonte: Autor

Os tempos de resposta obtidos demonstram a viabilidade da teoria do perigo para proteção de redes baseadas em SDN.

5

Considerações Finais

A possibilidade de gestão autonômica em redes SDN baseada em um sistema imune-inspirado utilizando o Algoritmo das Células Dendríticas, uma das abordagens mais promissoras da teoria do perigo, demonstrou-se aplicável em uma rede SDN.

Tendo em vista que a SDN já prover de características como gestão centralizada, que auxiliam no autogerenciamento, e, também, por ser programável possibilitou a adequação do modelo MAdPE-K para criação da arquitetura MAdPE-K/SDN, provendo função autonômica de autoproteção.

Este trabalho, apresentou uma arquitetura para a disponibilização de uma função autonômica de autoproteção para uma rede definida por software, utilizando uma abordagem do sistema imune-inspirado na teoria do perigo aplicando algoritmo das células dendríticas nos dispositivos de rede.

Os resultados alcançados nos experimentos realizados mostraram que é possível efetuar o monitoramento, análise, planejamento e execução, seguindo o modelo MAdPE-K, com baixo tempo de resposta. Por conseguinte, limitando a ação das etapas seguintes de um ataque.

Assim, pode-se afirmar que a abordagem da teoria do perigo é aplicável para autoproteção em redes definidas por software, viabilizando criação de mecanismos de segurança eficazes.

5.1 Contribuições

Este trabalho deve as seguintes principais contribuições: a definição de uma arquitetura, denominada MAdPE-K/SDN, para autoproteção em uma rede definida por *software* com base no modelo MAdPE-K, que utiliza conceitos da Teoria do perigo para predição de ataques.

Outras contribuições deste trabalho foram, através de prova de conceito, aplicar um sis-

tema com base em MAdPE-K/SDN em ambiente não simulado, demonstrando sua aplicabilidade em ambientes reais.

5.2 Trabalhos Futuros

Afim de dar continuidade a este trabalho serão realizadas novos experimentos em uma topologia com quantidade maior de elementos além de definir novas ações para mitigar o ataques como inibir o fluxo de pacotes identificado como anômalo.

Outros pontos sugeridos para trabalhos futuros são modelar um sistema de autoproteção contra ataques de negação de serviço e outros tipos de padrões de varredura; modelar mensagens entre os elementos para integrar com serviços de detecção de intrusão e definir políticas de mitigação com base nos tipos de ataque.

Referências

AHMAD, A.; IDRIS, N. B.; KAMA, M. N. Cloudids: Cloud intrusion detection model inspired by dendritic cell mechanism. *International Journal of Communication Networks and Information Security (IJCNIS)*, v. 9, n. 1, 2017. Citado na página 15.

AL-HAMMADI, Y.; AICKELIN, U.; GREENSMITH, J. Dca for bot detection. In: IEEE. *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*. [S.l.], 2008. p. 1807–1816. Citado na página 34.

ANANDITA, S. et al. Implementation of dendritic cell algorithm as an anomaly detection method for port scanning attack. In: *2015 International Conference on Information Technology Systems and Innovation (ICITSI)*. [S.l.: s.n.], 2015. Citado 7 vezes nas páginas 15, 16, 33, 49, 50, 53 e 54.

BEHRINGER, M. et al. *Autonomic Networking: Definitions and Design Goals*. [S.l.], 2015. Citado 4 vezes nas páginas 16, 25, 26 e 27.

BHUYAN, M. H.; BHATTACHARYYA, D.; KALITA, J. Surveying port scans and their detection methodologies. *The Computer Journal*, v. 54, n. 10, p. 1565–1581, 2011. Disponível em: <+http://dx.doi.org/10.1093/comjnl/bxr035>. Citado 3 vezes nas páginas 19, 20 e 21.

BOMFIM, L. H. d. S. *Um Serviço para Anonimização em Redes Definidas por Software*. 101 p. Tese (Doutorado) — UFS -Universidade Federal de Seripe, 2017. Citado na página 24.

BOMFIM, L. H. da S. et al. A systematic mapping about anonymization services for software-defined networking. *IEEE Latin America Transactions*, v. 15, n. 6, p. 1113–1120, June 2017. ISSN 1548-0992. Citado na página 24.

CASTRO, L. N. D.; ZUBEN, F. J. V.; KNIDEL, H. Artificial immune systems. In: . [S.l.]: Springer, 2007. Citado na página 15.

CHELLY, Z.; ELOUEDI, Z. A survey of the dendritic cell algorithm. *Knowledge and Information Systems*, v. 48, n. 3, p. 505–535, 2016. ISSN 02193116. Citado na página 29.

E. Haleplidis, E. et al. Software-Defined Networking (SDN): Layers and Architecture Terminology. *Internet Research Task Force (IRTF)*, p. 1–35, 2015. ISSN 2070-1721. Disponível em: <http://www.rfc-editor.org/info/rfc7426.https://tools.ietf.org/html/rfc7426>. Citado na página 22.

ENGHOLM, J. H. *Análise e Design Orientados a Objetos*. São Paulo: Novatec, 2013. Citado na página 45.

FEAMSTER, N.; REXFORD, J.; ZEGURA, E. The road to sdn: an intellectual history of programmable networks. In: ACM. *ACM SIGCOMM Computer Communication Review*. [S.l.], 2014. p. 87–98. Citado na página 21.

FERNANDES, D. A. et al. Applications of artificial immune systems to computer security: A survey. *Journal of Information Security and Applications*, Elsevier, v. 35, p. 138–159, 2017. Citado na página 15.

FLOODLIGHT. *Floodlight OpenFlow Controller -Project Floodlight*. 2014. http://www.projectfloodlight.org/floodlight/ p. Disponível em: <http://www.projectfloodlight.org/floodlight/>. Citado na página 22.

FOUNDATION, O. N. Software-defined networking: The new norm for networks. In: *ONF White Paper*. [S.l.: s.n.], 2012. Citado 2 vezes nas páginas 21 e 23.

GADGE, J.; PATIL, A. A. Port scan detection. In: *Proceedings of the 2008 16th International Conference on Networks, ICON 2008*. IEEE, 2008. p. 1–6. ISBN 9781424438051. ISSN 1556-6463. Disponível em: <http://ieeexplore.ieee.org/document/4772622/>. Citado na página 20.

GREENSMITH, J.; AICKELIN, U. Dendritic cells for syn scan detection. In: ACM. *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. [S.l.], 2007. p. 49–56. Citado 7 vezes nas páginas 15, 16, 31, 32, 33, 49 e 53.

GREENSMITH, J.; AICKELIN, U. The deterministic dendritic cell algorithm. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v. 5132 LNCS, p. 291–302, 2008. ISSN 03029743. Citado na página 31.

GREENSMITH, J.; AICKELIN, U.; CAYZER, S. Introducing Dendritic Cells as a Novel Immune-Inspired Algorithm for Anomoly Detection. *Proceedings of the 4th International Conference on Artificial Immune Systems (ICARIS2005), Lecture Notes in Computer Science 3627, Banff, Canada*, p. 153–167, 2005. Disponível em: <http://ima.ac.uk/papers/greensmith2005a.pdf>. Citado 3 vezes nas páginas 31, 33 e 54.

GREENSMITH, J.; AICKELIN, U.; TWYCROSS, J. Articulation and clarification of the dendritic cell algorithm. In: SPRINGER. *International Conference on Artificial Immune Systems*. [S.l.], 2006. p. 404–417. Citado 5 vezes nas páginas 11, 30, 32, 49 e 50.

HASHIM, F.; MUNASINGHE, K. S.; JAMALIPOUR, A. Biologically inspired anomaly detection and security control frameworks for complex heterogeneous networks. *IEEE Transactions on Network and Service Management*, IEEE, v. 7, n. 4, p. 268–281, 2010. Citado na página 14.

HORN, P. Autonomic computing: Ibm\'s perspective on the state of information technology. IBM, 2001. Citado na página 25.

IBM. Autonomic Computing White Paper: An Architectural Blueprint for Autonomic Computing. *IBM White Paper*, n. June, p. 34, 2005. ISSN 1944-8244. Disponível em: <http://www-03.ibm.com/autonomic/pdfs/ACBlueprintWhitePaperV7.pdfhttp://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:An+architectural+blueprint+for+autonomic+computi>. Citado 3 vezes nas páginas 14, 28 e 29.

International Organization for Standardization. *International Standard ISO/IEC 27000*. 2014. Disponível em: <https://www.iso.org/standard/63411.html>. Citado na página 19.

JANG-JACCARD, J.; NEPAL, S. A survey of emerging threats in cybersecurity. *Journal of Computer and System Sciences*, v. 80, n. 5, p. 973 – 993, 2014. ISSN 0022-0000. Special Issue on Dependable and Secure ComputingThe 9th {IEEE} International Conference on Dependable, Autonomic and Secure Computing. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0022000014000178>. Citado na página 17.

KEPHART, J. O.; CHESS, D. M. The vision of autonomic computing. *Computer*, IEEE, v. 36, n. 1, p. 41–50, 2003. Citado 4 vezes nas páginas 14, 25, 26 e 27.

KHONDOKER, R. et al. Feature-based comparison and selection of software defined networking (sdn) controllers. In: *Computer Applications and Information Systems (WCCAIS), 2014 World Congress on*. [S.l.: s.n.], 2014. p. 1–7. Citado na página 22.

KIM, H.; FEAMSTER, N. Improving network management with software defined networking. In: IEEE. *Communications Magazine*. [S.l.], 2013. p. 114–119. Citado na página 22.

KREUTZ D.; RAMOS, F.; VERISSIMO, P. E.; ROTHENBERG C.AND AZODOLMOLKY, S. U. S. E. Software-defined networking: A comprehensive survey. In: IEEE. *Proceedings of the IEEE*. [S.l.], 2015. p. 14–76. Citado na página 22.

LARMAN, C. *Utilizando UML e padrões*. [S.l.]: Bookman Editora, 2002. Citado 2 vezes nas páginas 45 e 46.

LYON, G. F. *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. USA: Insecure, 2009. ISBN 0979958717, 9780979958717. Citado na página 20.

MATZINGER, P. Tolerance, danger, and the extended family. *Annual review of immunology*, Annual Reviews 4139 El Camino Way, PO Box 10139, Palo Alto, CA 94303-0139, USA, v. 12, n. 1, p. 991–1045, 1994. Citado 5 vezes nas páginas 27, 28, 29, 33 e 37.

MIHAI-GABRIEL, I.; PATRICIU, V. V. Biologically inspired risk assessment in cyber security using neural networks. In: *2014 10th International Conference on Communications (COMM)*. [S.l.: s.n.], 2014. p. 1–4. Citado na página 37.

NUNES, B. A. A. et al. A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks. *IEEE Communications Surveys & Tutorials*, v. 16, n. 3, p. 1617–1634, 2014. ISSN 1553-877X. Disponível em: <http://arxiv.org/abs/1406.0440http://ieeexplore.ieee.org/document/6739370/>. Citado na página 24.

OLIVEIRA, D. D. de; SALGUEIRO, R. J. d. B.; MORENO, E. D. Predizendo o perigo: Alerta antecipado contra a propagação de worms utilizando o algoritmo das células dendríticas. *SBRC - XVIII Workshop de Gerência e Operação de Redes e Serviços - WGRS*, p. 105–118, 2013. Citado 7 vezes nas páginas 16, 34, 35, 39, 49, 50 e 53.

OPENDAYLIGHT. *OpenDaylight - Open Source SDN Platform*. 2015. https://www.opendaylight.org// p. Disponível em: <https://www.opendaylight.org/>. Citado na página 22.

RAWAT, S.; SAXENA, A. Danger theory based syn flood attack detection in autonomic network. In: ACM. *Proceedings of the 2nd international conference on Security of information and networks*. [S.l.], 2009. p. 213–218. Citado na página 15.

RYU SDN Project Team. *RYU SDN Framework*. 2016. 455 p. Disponível em: <https://osrg.github.io/ryu-book/en/Ryubook.pdf>. Citado na página 22.

SAHOO, K. S. et al. A Comprehensive Tutorial on Software Defined Network. In: *Proceedings of the International Conference on Advances in Information Communication Technology & Computing - AICTC '16*. New York, New York, USA: ACM Press, 2016. p. 1–6. ISBN

9781450342131. Disponível em: <http://dl.acm.org/citation.cfm?doid=2979779.2983928>. Citado na página 23.

SCOTT-HAYWARD, S.; NATARAJAN, S.; SEZER, S. A survey of security in software defined networks. *IEEE Communications Surveys and Tutorials*, v. 18, n. 1, p. 623–654, 2016. ISSN 1553877X. Disponível em: <http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=7150550http://ieeexplore.ieee.org/document/71505>. Citado na página 24.

SCOTT-HAYWARD, S.; O'CALLAGHAN, G.; SEZER, S. Sdn security: A survey. In: IEEE. *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*. [S.l.], 2013. p. 1–7. Citado na página 24.

SEZER, S. Are we ready for sdn? implementation challenges for software-defined networks. In: IEEE. *Communications Magazine*. [S.l.], 2013. p. 36–43. Citado na página 21.

SEZER, S. et al. Are we ready for SDN? Implementation challenges for software-defined networks. *IEEE Communications Magazine*, v. 51, n. 7, p. 36–43, jul 2013. ISSN 0163-6804. Disponível em: <http://ieeexplore.ieee.org/document/6553676/http://arxiv.org/abs/1406.0440>. Citado na página 25.

SHAMSHIRBAND, S. et al. Co-fais: cooperative fuzzy artificial immune system for detecting intrusion in wireless sensor networks. *Journal of Network and Computer Applications*, Elsevier, v. 42, p. 102–117, 2014. Citado na página 15.

SHIREY, R. Rfc 4949–internet security glossary.[sl]: Version, 2007. *Citado na*, p. 32. Nenhuma citação no texto.

SILVA, G. C.; CAMINHAS, W. M.; ERRICO, L. de. Dendritic cell algorithm applied to ping scan investigation revisited: Detection quality and performance analysis. *IEEE Transactions on Emerging Topics in Computational Intelligence*, v. 1, n. 4, p. 236–247, Aug 2017. Citado na página 15.

The POX Controller Team. *The POX Controller*. 2016. Disponível em: <https://github.com/noxrepo/pox>. Citado na página 22.

The Trema Controller Team. *Trema - Full-Stack OpenFlow Framework in Ruby and C*. 2016. Disponível em: <http://trema.github.io/trema/>. Citado na página 22.

WENDONG, W. et al. Autonomicity design in OpenFlow based Software Defined Networking. In: *2012 IEEE Globecom Workshops, GC Wkshps 2012*. [s.n.], 2012. p. 818–823. ISBN 9781467349413. Disponível em: <https://195.134.65.236/IEEE{_}Globecom{_}2012/papers/p818-wendo>. Citado 2 vezes nas páginas 36 e 37.

YAN, Q. et al. Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges. *Communications Surveys Tutorials, IEEE*, PP, n. 99, p. 1–1, 2015. ISSN 1553-877X. Citado 2 vezes nas páginas 16 e 36.

YANG, H. et al. A survey of artificial immune system based intrusion detection. *The Scientific World Journal*, Hindawi Publishing Corporation, v. 2014, 2014. Citado na página 15.